

ALGORITHMS IN RIGIDITY THEORY WITH
APPLICATIONS TO PROTEIN FLEXIBILITY AND
MECHANICAL LINKAGES

ADNAN SLJOKA

A DISSERTATION SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN MATHEMATICS AND STATISTICS
YORK UNIVERSITY,
TORONTO, ONTARIO

August 2012

ALGORITHMS IN RIGIDITY THEORY
WITH APPLICATIONS TO PROTEIN FLEXIBILITY
AND MECHANICAL LINKAGES

by Adnan Sljoka

a dissertation submitted to the Faculty of Graduate Studies of York University in
partial fulfillment of the requirements for the degree of

Doctor of Philosophy

© 2012

Permission has been granted to: a) YORK UNIVERSITY LIBRARIES to lend or sell copies of this thesis in paper, microform or electronic formats, and b) LIBRARY AND ARCHIVES CANADA to reproduce, lend, distribute, or sell copies of this thesis anywhere in the world in microform, paper or electronic formats and to authorize or procure the reproduction, loan, distribution or sale of copies of this thesis anywhere in the world in microform, paper or electronic formats.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the authors written permission.

Abstract

A natural question was asked by James Clark Maxwell: Can we count vertices and edges in a framework in order to make predictions about its rigidity and flexibility? The first complete combinatorial generalization for (generic) bar and joint frameworks in dimension 2 was confirmed by Laman in 1970. The $6|V| - 6$ counting condition for 3-dimensional molecular structures, and a fast ‘pebble game’ algorithm which tracks this count in the multigraph have led to the development of the program FIRST for rapid predictions of the rigidity and flexibility of proteins.

In this thesis we develop the mathematical models, algorithms and theory using the concepts from combinatorial rigidity theory that have various important applications in protein science and mechanical engineering. Using our extensions of the pebble game algorithm, we have developed a novel protein hinge prediction algorithm. The hinge predictions are tested on numerous hinge bending proteins. We have also introduced several rigidity-based allostery models with corresponding allostery-detection algorithms that can detect transmission of rigidity (change in shape) between distant sites. Various examples and theoretical results on the rigidity-based allosteric communication will be provided. We will apply our algorithm on some allosteric proteins, including the important class of signalling proteins known as G-protein couple receptors. The results obtained show that rigidity-based modelling of allostery and our algorithms are promising tools in studying allostery in proteins. We

also present a novel FIRST-ensemble method which extends FIRST to give a prediction of rigidity/flexibility of structural protein ensembles, and when combined with ensemble solvent accessibility data, it gives a good prediction of hydrogen-deuterium exchange. In the area of mechanical linkages, we have developed novel techniques and a pinned version of the pebble game algorithm that can decompose linkages into important d -Assur graphs.

To my parents

Acknowledgments

I would like to express my deep sense of gratitude and heartfelt thanks to my supervisor, Professor Walter J. Whiteley. Without his invaluable guidance and countless support this work would have been impossible. I feel very grateful and privileged to have worked with Professor Whiteley and to have him as my supervisor during my graduate studies. He has taught me a lot about research, to not hesitate to ask questions and make new conjectures, and his friendly teaching approach, enthusiasm and expertise on many subjects has made my research and studies a lot more enjoyable. He has always been generous with his time and made himself available during the countless (sometimes very long) stimulating meetings, and has made numerous suggestions on this thesis. The nurturing spirit and support from Professor Whiteley during the difficult times will always be remembered. I am indebted to Professor Whiteley for nourishing my personal, intellectual and academic growth, from which I will benefit for a long time.

As my work spanned different fields, this resulted in several critical collaborations. I would like to express special thanks to Professor Derek Wilson, who was also on my supervisory committee, for his patience and numerous helpful discussions about protein flexibility and how it relates to my work. I am also grateful for many

fruitful discussions and e-mail exchanges with Professor Offer Shai and his encouragement to pursue and extend my work to problems and applications in mechanical engineering.

I also want to thank Professor Huaxiong Huang and Professor Jorg Grigull for taking their time to serve on my supervisory committee. Their input, advice and connections to new projects and applications was very valuable.

Thanks also to Professor Mike Thorpe, Professor Suprakash Datta and Professor Ada Chan for agreeing to be on my examining committee. I want to thank Professor Thorpe for his expertise and several valuable discussions about my work and topics on protein flexibility.

I benefitted a lot from other students and friends who were very helpful throughout my studies. I especially thank Alexandr Bezginov for his valuable help in the initial implementations, with Pymol scripts and continued interest in the rigidity applications. I want to thank Dr. Naveen Vaidya, a former graduate student, for his assistance and new collaborative projects and his friendship over the years. I also thank Dr. Bernd Schulze for many discussions on the rigidity and his friendship and interest in collaboration. I also want to thank Primrose Miranda for her very helpful administrative support.

I want to give a special thanks to Cora for all the support and encouragement through the long writing process.

Last but not least, I want to thank my parents for their love and continued support. They were always there for me and made many sacrifices through this long and sometimes challenging journey. Puno hvala na svemu što ste mi pružili, za podršku i veliko stpljenje, voli vas vaš sin. I also thank my brother for his support (voli te tvoj brat) and his family for their understanding.

Contents

Abstract	iv
Acknowledgments	vii
1 Introduction	1
1.1 Overview	1
1.2 Protein rigidity and flexibility	4
1.2.1 Protein flexibility is related to protein function	5
1.3 Protein flexibility can be studied using Rigidity Theory	8
1.3.1 FIRST method overview	8
1.4 Thesis Outline	15
2 Rigidity Theory: Background and Searching for the Counts	20
2.1 Overview	20
2.2 Rigidity Theory – introduction	21
2.2.1 Basic concepts	21
2.2.2 Definitions, Rigidity and Infinitesimal Rigidity	27
2.3 Counting for rigidity in plane graphs and Laman’s Theorem	39
2.3.1 Counting is not sufficient in dimension 3	45
2.4 Counting in Body-Bar and Molecular Structures	48
3 The Pebble Game Algorithm and Relevant Regions	61
3.1 Overview	61
3.2 The Pebble Game Algorithm	62
3.2.1 Pebble game illustration and properties	70
3.3 The Relevant Regions	81
3.3.1 Quantifying relative flexibility of the core	82
3.3.2 Relevant region detection	89

3.3.3	Properties of the Relevant regions	101
4	Hinge motion predictions	103
4.1	Overview	103
4.2	Hinge motions and background	105
4.3	Hinge motion detection via Relevant Regions	110
4.4	Hinge-Prediction algorithm for proteins	124
4.5	Application of hinge-prediction algorithm on proteins	129
4.6	Concluding remarks and future work	152
5	Rigidity model of Protein Allostery	157
5.1	Overview	157
5.2	Allostery background	159
5.3	A Rigidity approach to allostery	165
5.4	Rigidity models of allostery	175
5.5	Methods and Algorithms to detect rigidity-based allosteric communication	181
5.5.1	Algorithms	182
5.5.2	Examples	190
5.5.3	Properties of Allostery type 2	203
5.5.4	Cyclic communication in Allostery type 2	206
5.6	Applications to protein allostery	214
5.6.1	Rigidity-based allostery in calmodulin	214
5.6.2	Applications to G-Protein Coupled Receptors	222
5.7	Concluding remarks and future work	252
6	Predictions of rigidity and flexibility of Protein ensembles and Hydrogen-Deuterium exchange	260
6.1	Overview	260
6.2	Introduction	262
6.2.1	Extending FIRST to Protein ensembles	262
6.2.2	Hydrogen/Deuterium exchange is dependent on Rigidity and Solvent Accessibility	267
6.3	Methods and Algorithms	270
6.3.1	FIRST-ensemble rigidity/flexibility predictions	270
6.3.2	FIRST-ensemble algorithm	276

6.3.3	Solvent Accessibility Calculations	282
6.3.4	Combining FIRST-ensemble and Solvent Accessibility on ensemble as a Predictor of HD-exchange	284
6.4	Results and Discussions	287
6.4.1	Experimental HDX profile of Sso AcP	287
6.4.2	FIRST rigidity/flexibility predictions using individual NMR models	290
6.4.3	FIRST-ensemble rigidity/flexibility predictions	292
6.4.4	Solvent Accessibility ensemble data	299
6.4.5	Combined rigidity and solvent accessibility predictions of HDX	304
6.5	Concluding remarks and future work	308
7	Decompositions of Linkages, Assur graphs and the pinned pebble game algorithm	311
7.1	Overview	311
7.2	Introduction	313
7.3	Decomposition of Pinned Directed Graphs	320
7.3.1	Strongly connected component decomposition	321
7.3.2	Equivalent orientations of a graph	324
7.4	Decomposition of Pinned d -isostatic Graphs	327
7.4.1	Pinned d -isostatic graphs and pinned rigidity matrix	327
7.4.2	Directed graph decomposition of the pinned rigidity matrix	332
7.4.3	d -Assur graphs and minimal components	338
7.4.4	Summary Assur Decomposition	339
7.4.5	Pinned d -counts and insufficiency of d -directions	340
7.5	Tracing Motions in Linkages through Assur Decompositions	349
7.5.1	Motions generated by removing an edge: d -Assur vs strongly d -Assur graphs	349
7.5.2	Vertex removal	354
7.6	Pinned pebble game algorithm and Assur decomposition algorithm	356
7.6.1	Pinned 2-dimensional pebble game	359
7.6.2	2-Assur Decomposition Algorithm	366
7.6.3	Tracking mobility of vertices using the pebble game	369
7.7	Concluding Remarks and Future Work	373
8	Conclusion	376

References 378

Chapter 1

Introduction

“Look deep into nature, and then you will understand everything better.”

– Albert Einstein

1.1 Overview

Rigidity theory has a rich history which can be traced back to Leonhard Euler, Baron Augustin-Louis Cauchy, James Clerk Maxwell and a host of other mathematicians and engineers. Euler (1766) conjectured that “A closed spatial figure allows no changes, as long as it is not ripped apart” [71]. Using today’s terminology, a closed spatial figure is a closed polyhedral surface made up of rigid polygonal plates that are hinged along the edges where plates meet. Maxwell [125] (1864) was exploring whether structures were stable or unstable (deformable), and a host of other engineers worked on different applications, such as determining the structural stability of different truss configurations in bridges and in mechanical linkages [72].

Rigidity theory for plane bar and joint frameworks was already well advanced in the later half of the nineteenth century. Henneberg in the early twentieth century summarized previous work and introduced important inductive techniques that are

used to construct 2-dimensional bar and joint frameworks starting from the basic building blocks [77]. Henneberg's inductive techniques, together with other more recent techniques, are still widely used, particularly in proofs and rigidity analysis of frameworks [203]. While a school of work in the rigidity theory continued for decades in the former Soviet Union [70], overall the best discoveries of the nineteenth century in the rigidity theory of bar and joint frameworks was, for the most part forgotten for at least the first half of the twentieth century.

The modern era and significant advancement in rigidity theory begins with the Dutch mathematician Gerard Laman. Laman's theorem (1970) [111] gives a beautiful and complete combinatorial characterization of rigidity and flexibility of bar and joint frameworks in dimension 2. In combinatorial characterization of rigidity one can simply count the vertices and edges (constraints) in a graph and its subgraphs to determine the rigidity and flexibility of a corresponding framework. Even though Maxwell and various others, including mechanical engineers [72], were counting vertices and edges in a graph to analyze rigidity, Laman's theorem gives a first rigorous characterization. Around the same time mathematicians and engineers began to interact and further revive and advance the subject [32].

Since that time, there has been an evolving and increased interest in this rich and fascinating subject, and our understanding and characterization of rigid and flexible structures has tremendously grown over the past three decades. One result that will be important in this thesis is the Molecular Theorem in rigidity theory, which offers a remarkable combinatorial counting result (like Laman's Theorem) for determining the rigidity and flexibility of molecular structures, and is used in fast computational predictions of rigidity and flexibility of macromolecules, such as proteins. The Molecular Theorem, which was conjectured about 30 years ago by Tay and Whiteley [100] was just recently proven [203].

Mathematicians and researchers from many different fields (such as biochemistry, computer science, physics, material science, robotics, engineering) are continuing to work on problems (including some very old unsolved problems) and applications related to rigidity theory. The number of different applications that rely on the rigidity theory is rapidly expanding. Some of these include: protein and glass network flexibility predictions [109, 187, 188], sensors and communication [4, 43], computer aided design (CAD) [205], decomposition of linkages in mechanical engineering [163, 174], and various others.

This thesis can be divided into two main strands, which includes both theoretical work and various applications. First, we will develop mathematical models, algorithms and corresponding theory using the concepts and techniques from the (combinatorial) rigidity theory with the underlying motivation that these algorithms and methods have important applications to problems in protein science and mechanical engineering. We will then directly apply and illustrate these methods and algorithms to problems such as: hinge predictions of hinge bending proteins (Chapter 4), the elusive allosteric communication in proteins (Chapter 5), prediction of rigidity/flexibility of structural protein ensembles together with computational predictions of experimental measurement - the Hydrogen/Deuterium exchange (Chapter 6), and also in important decompositions of linkages from mechanical engineering (Chapter 7).

Throughout our work in this thesis, we will almost entirely focus on the combinatorial characterization of rigidity (also referred to as graph rigidity) of the structures, as combinatorial (i.e. counting) results lead to fast and practical algorithms, such as the 'pebble game algorithm'. The pebble game algorithm will be central to our methods and will be discussed in detail (Chapter 3), and is the main component

of the program FIRST (Floppy Inclusion and Rigid Substructure Topography) [48], which gives fast predictions of rigidity and flexibility of proteins.

The majority of our algorithms and methods in this thesis will be applied to protein applications (which will form chapters 4 - 6). The development of the algorithms will often involve various extensions of the pebble game algorithm, which allows us to extract additional useful information. Since one of the major objectives in this thesis is applications of our algorithms to problems related to protein rigidity and flexibility, in this introductory chapter, we will provide some general motivational background on importance of studying protein rigidity and flexibility. As we will extensively use the program FIRST, whose rigidity and flexibility predictions will often form the initial input to our algorithms, in this chapter we will also briefly introduce the methodology of FIRST, leaving full details to the provided references.

At the end of this chapter, we will offer a brief summary of the chapters and a short description of the main problems (chapters 4 - 7) that will be addressed in this thesis.

1.2 Protein rigidity and flexibility

Protein basics

Proteins (from the Greek “protos” meaning “of primary importance”) are the most versatile macromolecules which perform crucial functions in essentially all biological processes. Proteins function as catalysts, they transport and store other molecules such as oxygen, they provide immune protection, and are also responsible for carrying out important transduction signals in and out of the cells, among many other biologically significant functions [15]. These macromolecules are involved in numerous molecular interactions: with other proteins, DNA, RNA, and small molecules (drugs)

[15]. Proteins are composed of sequences drawn from twenty amino acids, sometimes referred as the building blocks of life, and each protein has its own specific sequence of amino acids (primary structure).

The three-dimensional structure (shape) of a protein is encoded in its primary amino acid sequence. Ideally one would like to have a computer program, which takes an amino acid sequence of a particular protein and deduces or predicts its 3-dimensional structure (specific spatial positions of atoms). In spite of considerable efforts in many fields such as bioinformatics and computational biology, the *protein folding problem* is still considered as one of the most important intellectual challenges in molecular biology [15]. Because of the direct link between protein's function and its three-dimensional structure, finding the solution to the "Folding problem" is an essential task. Fortunately many protein structures can be determined (at atomic resolution) by experimental methods such as x-ray crystallography or nuclear magnetic resonance (NMR) techniques. The solved structures are deposited into the protein data bank (PDB) [13].

A quick visualization of the protein's 3-dimensional structure using any popular molecular visualization software such as PYMOL [143], reveals their immense complexity (see Figure 1.1). Proteins typically contain thousands of atoms, and despite their complexity, they are fairly compact. They commonly form motifs with regular periodicity, such as alpha helices and beta sheets.

1.2.1 Protein flexibility is related to protein function

It is well known that protein structural flexibility and dynamics is as critical for protein function as its 3D-structure [8, 15, 35, 107]. Following the basic principle: *if you know how it moves, you can better interpret how it works*, the knowledge of protein flexibility offers a direct connection between its structure and function. Accurate

measurements of flexibility and dynamics of proteins can help us to interpret the relationship between structure and function [15, 25, 33, 67, 109, 218].

The secondary structural elements (such as alpha helices or beta sheets) of domains as well as entire domains (stable regions) undergo movements in space. These movements involve either fluctuations of individual atoms or collective rigid-body motions of a group of atoms. Proteins need to be rigid enough to maintain their shape and flexible enough to perform biological functions. Some diseases are linked to key proteins adopting non-native conformations (i.e. incorrect misfolded shapes) which lead to creation of overly rigid and indestructible complexes (i.e. ‘amyloid and prion diseases’ such as Alzheimers, Parkinsons, Mad Cow Disease and its human variant Creutzfeldt-Jakob Disease, etc.) [15, 55, 105, 144, 183]. In contrast, there are diseases such as cystic fibrosis, that due to a single point mutation cause a mutant protein to become too flexible, which subsequently does not attain enough shape (or is too slow to take shape) causing it to be degraded much more rapidly and prematurely (so it never gains function) compare to the more rigid and stable non-mutant protein [121]. Being able to give fast predictions of flexible and rigid regions in the proteins and its dynamics is an important area of research in computational biology, and has significant biological implications in medicine and applications such as drug design.

Determining flexible and rigid regions in a protein and predicting its motions is a complex task. This is not only due to the proteins complex structures but also due to their wide range of internal motions that span length-scales from 0.01 \AA to 10 \AA , and timescales from 10^{-15} (local vibrations) to $>1 \text{ s}$ [35] (large scale motions). A wide range of experimental data (multiple configurations, NMR ensembles, hydrogen/deuterium exchange data) can provide some experimental insight [64, 87, 139]. In addition to experimental methods, computational simulations have also facilitated

enormous strides on this topic and several new computational techniques have been introduced over the last two decades [18, 49, 108, 109, 126, 132, 167].

A traditional computational approach, which has been widely used in obtaining detailed information on the protein mobility and flexibility is molecular dynamics (MD) simulation. In this technique, one solves the Newton's classical equations of motions $F = ma$ by forward integration in time. The limitation of using MD simulations is that biochemical events and functionally important protein motions (such as the hinge motions between domains [62] - see Chapter 4) often take place on order of micro- (10^{-6}) to millisecond timescales (10^{-3}) which are far beyond the reach of MD simulations (each time step in molecular dynamics simulation is on the femtosecond (10^{-15}) time scale) [102]). This requires a prohibitive amount of computational power to model and the computational time needed to reach these large scale motions is (normally) beyond practical wide-range application. Some speedups of the simulation can be achieved using coarse-graining and parallelized MD methods with programs such as NAMD [131], and further improvements can be made using massively parallelized MD runs via costly special-purpose commodity computer clusters, such as Anton [168]. Even with the speedups it often does not lead to practical use.

Improvements to protein structure determination techniques have rapidly increased the number of solved 3-dimensional structures (the current count in the PDB exceeding 80000 structures). To add to this, one can download the structures of complete viral capsids (i.e. outer shell of the virus), which typically contain hundreds of proteins with several hundred thousands atoms.

Given this rapid growth in the number of entries in the protein data bank and the size of available protein structures, combined with the severe limitation of computational power and resources needed to study protein flexibility and dynamics with traditional methods, there is a tremendous need to develop faster computational

methods. One such computationally fast new method is FIRST [48, 109], along with its extension FRODA [199], the geometric simulation algorithm, which uses the output of FIRST as a pre-processing step to model the initial motions of the protein.

1.3 Protein flexibility can be studied using Rigidity Theory

We now give a brief introduction to the method FIRST. Here we will use the informal rigidity and flexibility terminology. More formal definitions will be provided in Chapter 2.

1.3.1 FIRST method overview

While considerable computational resources are necessary to probe the flexible and rigid regions with MD simulations, the program FIRST can predict the rigid clusters (i.e. rigid regions in a protein) and flexible connections in a matter of seconds on a typical processor. Computationally, FIRST uses the pebble game algorithm (see Chapter 3), which performs the rigidity analysis by constraint counting given in the Molecular Theorem (see Chapter 2) on the underlying molecular multigraph (constraint network) by matching the degrees of freedom with various biochemical constraints (see below for more details). The division of a protein into rigid clusters using FIRST is referred to as *rigid cluster decomposition* (Figure 1.1).

In numerous studies it has been thoroughly demonstrated with experimental evidence that FIRST gives accurate predictions of flexibility and detection of rigid clusters in proteins [67, 82, 83, 109] and RNA [56]. For instance, in one of the early papers on FIRST [109], the authors provide an elegant and visually rich illustration of the rigidity/flexibility prediction using FIRST on the case study HIV protease,

which is a major inhibitory drug target and an important protein in HIV replication process [160]. The authors performed FIRST rigidity/flexibility prediction on HIV protease (among other proteins) with and without the drug (inhibitor) bound. FIRST correctly predicted (matching the experimental evidence) the flexibility in the binding region (flexible binding loops - flaps) with no drug bound, and with the drug bound, the HIV protease had significant rigidification in the important binding loops, leading to the desired loss of function.

Due to the efficiency and the speed of the pebble game algorithm, FIRST becomes particularly attractive in the analysis of very large systems. In fact, the original motivation in the development of the pebble game algorithm was to analyze the rigidity/flexibility of the covalent glass networks, which can contain millions of atoms [92]. FIRST has been successfully applied on large molecular structures such as the ribosomal complex [57] and even viral capsids [84], which typically contain hundreds of proteins with several hundred thousand atoms. See Figure 1.1 for a typical output of FIRST showing the rigid cluster decomposition which is mapped (coloured) back on the protein structure and viewed with a popular molecular visualization software PYMOL [143].

As we will use FIRST in various forms in several key chapters of this thesis, we now provide a slightly more detailed summary.

Starting with the 3-dimensional protein structure (a snapshot containing atomic coordinates, i.e. PDB file), FIRST generates a constraint multigraph [84, 207], where the molecule is viewed as a body-hinge (Chapter 2) engineered structure of fixed units (atoms or bodies with their bond angles as rigid units, bonds as potential hinges, allowing only dihedral bond rotations between the atoms), plus other molecular (non-covalent interactions) constraints extracted from the local geometry (hydrogen bonds,

hydrophobic contacts or tethers). High frequency motions (i.e. bond-length vibrations) are neglected. This is a reasonable modelling assumption, as single covalent bond lengths over longer time scales are (essentially) invariant; for example, a length of a single covalent bond between a pair of carbon atoms will vary less than a percent from its equilibrium value of ca. 1.53 \AA [74]. Double bonds are modelled as a single rigid unit. The combinatorial rigidity theoretical results on these structures are given in Chapter 2.

In the constraint multigraph, atoms are represented by vertices and edges represent the distance constraints corresponding to covalent bonds, hydrogen bonds and hydrophobic interactions. The strength of each potential hydrogen bond is calculated based on its local donor atom–hydrogen–acceptor atom (angular and distance) geometry (for instance, stronger interactions are attained when these atoms are nearly collinear, see [82, 109, 200] for details of how FIRST models hydrogen bonds and hydrophobic contacts). Once all the constraints are considered and the user has selected a hydrogen bond energy cutoff value, FIRST uses a pebble game algorithm to rapidly decompose a graph into rigid clusters and flexible regions [109].

The hydrogen bond cutoff value is an adjustable parameter and is meant to discard all potential hydrogen bonds with bad geometry. Only hydrogen bonds whose energy is less than (i.e. more favourable) than the chosen cutoff value are included as constraints in the analysis, and other hydrogen bonds are excluded. The default hydrogen bond cutoff on the web interface version of FIRST is set at -1.0 kcal/mol , however various different cutoff values are used in the literature [82, 109, 200], providing good predictions with experimental evidence. In general, the cutoff value should be chosen so as to form and test a specific hypothesis about the rigidity and flexibility of the specific protein [200]. One normally chooses a cutoff depending on the type of a problem and motion being studied and on availability of any previous experimental

evidence (see also discussion on dilution plot below). The main message is that there is no single ‘correct’ universal cutoff that works for all proteins and for all types of proteins motions. This is part of many previous and current investigations [109, 200] (see also section 6.3.1 for detailed discussion on the impact of the selected set of hydrogen bonds as constraints in the graph on FIRST analysis).

With the generated multigraph (constraint network), the FIRST model of a protein allows for an exact determination of the protein rigidity and flexibility. Each bond is categorized as either flexible (able to rotate) or rigid (non-rotatable). Constraints (bonds in the protein - edges in the graph) remove degrees of freedom, thus, the existence of a rigid cluster indicates that there is a sufficiently high number of constraints in those clusters (Chapter 2). In every rigid cluster, all bonds will be non-rotatable and all its atoms will move together as a single rigid body. Flexible regions are under-constrained, as they do not have enough constraints to further restrict their internal motions. Typical proteins will have one or several larger rigid clusters, which are connected by flexible regions ¹.

In addition to the rigid cluster decomposition, the pebble game algorithm enables FIRST to also detect various other useful properties, such as redundance in rigid regions i.e. presence of additional constraints in already rigid region. Redundance is an indication of extra robust rigidity, as small changes such as breaking of a hydrogen bond will likely not affect the rigidity of the rigid cluster.

Changes in the rigidity can be monitored by a gradual removal of hydrogen bonds one by one in the order of strength (i.e. weak hydrogen bonds are removed first) keeping all covalent and hydrophobic interactions intact, and then redoing the FIRST

¹Note that the flexibility predicted by FIRST corresponds to a ‘virtual’ (snapshot - infinitesimal) motion as a prelude to real (finite, continues) motion. Only the potential of motion (i.e. infinitesimal) is identified, but this corresponds to finite motions under certain ‘generic’ assumptions of the molecular framework. This idea of the snapshot, mathematically better known as “infinitesimal” rigidity, is discussed in Chapter 2.

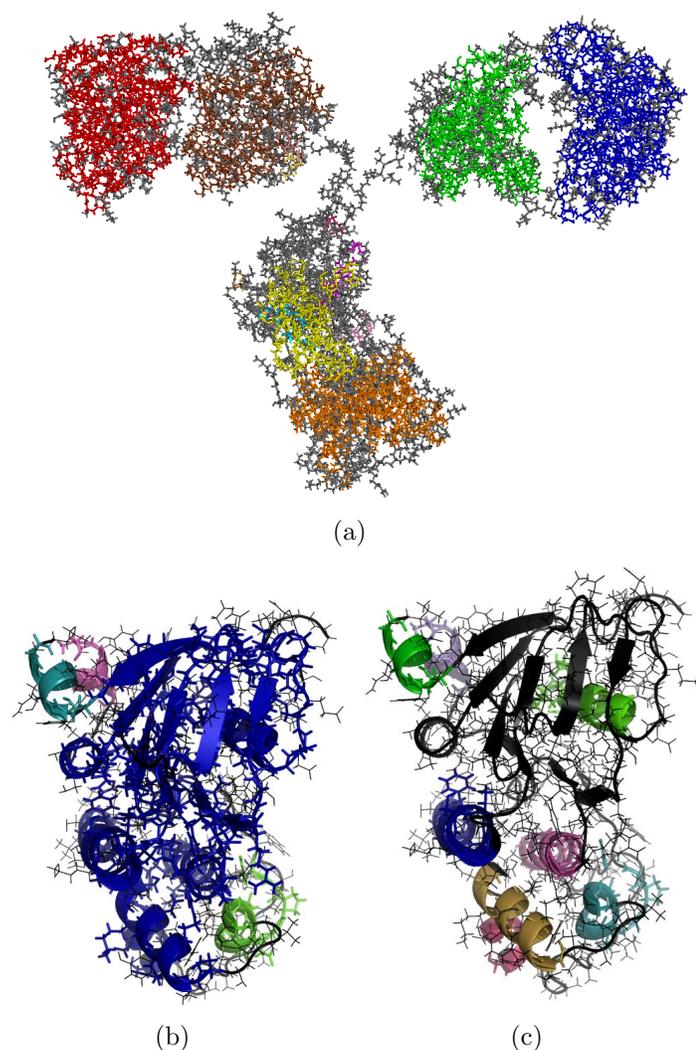


Figure 1.1: Illustration of the rigid cluster decomposition decomposition (RCD) obtained from FIRST [48] on two proteins. (a) RCD of immunoglobulin (pdb id: 1igt) in stick representation. Each coloured region represents a single rigid cluster (rigid body) where all bonds in each cluster are non-rotatable and can only move using trivial rigid body motions (i.e. translations and rotations). The connecting gray regions are flexible. In (a) and (b) the RCD of cystic fibrosis transmembrane conductance regulator (CFTR) nucleotide binding domain (NBD) (mutant form - pdb id: 2pzf, chain A) is shown in cartoon representation showing the secondary structures (α -helices and β -sheets) using hydrogen bond energy cutoffs -0.5 kcal/mol (b) and -1 kcal/mol (c). With a lower cutoff only strong hydrogen bonds remain and the protein is composed of smaller rigid clusters. The largest rigid cluster is coloured in blue. Black regions are flexible. Generated with Pymol [143].

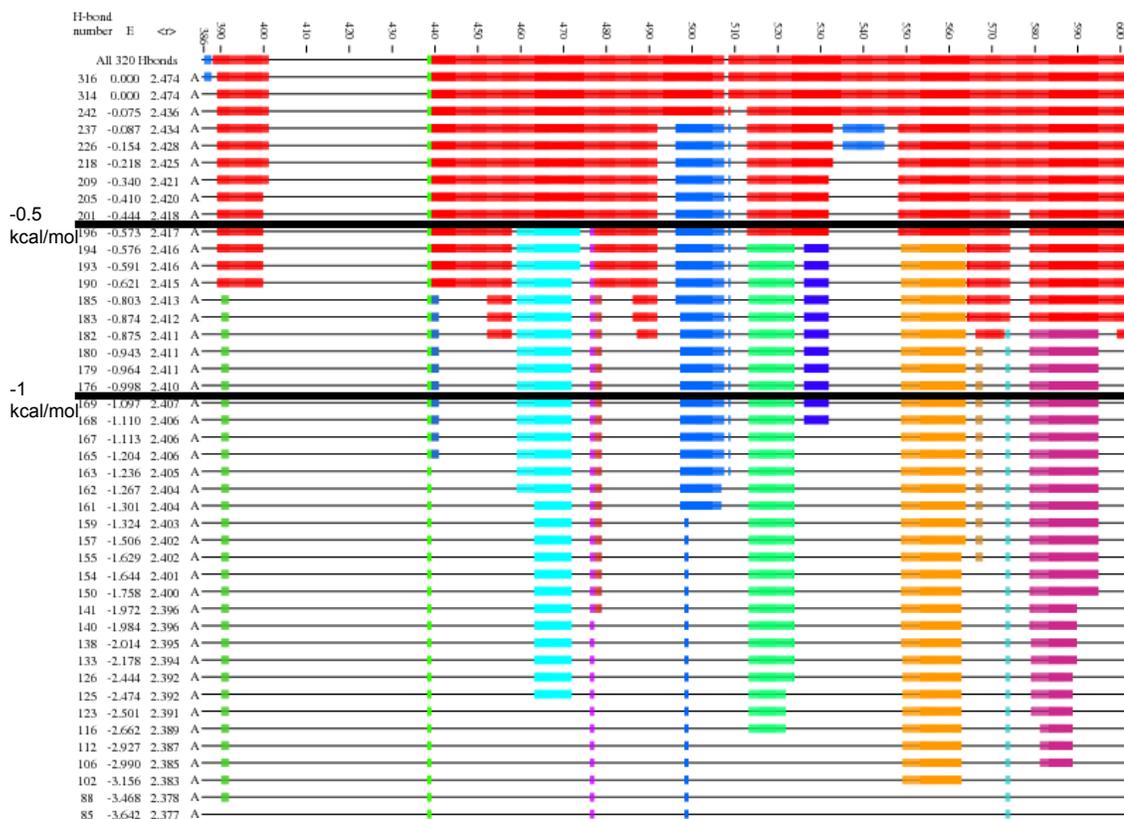


Figure 1.2: Dilution plot of cystic fibrosis transmembrane conductance regulator (CFTR) nucleotide binding domain (NBD) (mutant form - pdb id: 2pzf, chain A). Flexible regions are indicated with thin black lines, and rigid regions are indicated with coloured blocks. Initially with all potential hydrogen bonds included, almost the entire protein is rigid and composed of a single rigid region (red block), with the exception of residues near the N-terminal region of the protein. As energy of hydrogen bonds decreases (going down the dilution plot) more weak hydrogen bonds are broken and the larger rigid cluster breaks into several smaller rigid clusters. Observe that a number of rigid clusters remain rigid over a wide range of hydrogen bond energy cutoffs, particularly α -helices (see previous figure). The two thick black lines represent the two energy cutoffs, -0.5 kcal/mol and -1 kcal/mol, which are used to obtain the rigid cluster decomposition in Figure 1.1 (b) and (c), respectively.

(pebble game) on the resulting graph at each step identifying rigid and flexible regions. Breaking of hydrogen bonds from weakest to strongest essentially simulates thermal denaturation (unfolding) and has been previously used to detect a phase transition which defines the rigid percolation threshold that corresponds to the folded-unfolded

transition in protein folding [82, 83]. The change in rigidity can be visualized using the hydrogen bond dilution plot [109] (see also Chapter 4 and 6 for more details).

In Figure 1.2 a dilution plot of CFTR nucleotide binding domain protein is shown. In the dilution plot the horizontal axis represents the protein's primary structure. Flexible regions of the polypeptide chain correspond to thin black lines, and the regions corresponding to rigid clusters are shown with coloured blocks. Different colours are used to denote which residues belong to which rigid cluster. The vertical axis represents the dilution of the hydrogen bond constraints from the protein, with gradual progressive lowering of the energy cutoff. Every time a rigid cluster is broken into smaller rigid clusters, the columns on the left-hand side are updated and indicate: the total number of remaining hydrogen bonds, the energy of the hydrogen bond just broken in kcal/mol and the mean coordination number of the atoms over the entire protein at that step (average number of bonds per atom). Dilution is performed every time a hydrogen bond is removed to see how rigidity of the protein is changed, and frequently removal of some hydrogen bonds (i.e. redundant bonds) will not affect the rigid clusters, which would produce identical lines in the dilution plot. Only lines which correspond to the steps that change the rigid clusters are displayed. As most weak hydrogen bonds are broken (i.e. energy cutoff is lowered), the protein breaks down into several rigid clusters, where some residues that are spatially adjacent but widely separated along the backbone are part of the same rigid cluster. This breaking of rigid clusters and introduction of more flexibility at a given cutoff gives a step-like appearance to the dilution plot. The dilution plot is also useful in selecting the hydrogen bond cutoff value.

Referring to the dilution plot in Figure 1.2, we note that up to -0.5 kcal/mol cutoff, most of the protein remains part of a single rigid region and at -1 kcal/mol cutoff, the protein breaks into several smaller rigid clusters (see Figure 1.1). These

smaller rigid clusters are mostly composed of separate α -helices, and four helices retain their rigidity to lower cutoff values. Detailed study of impact of cutoff values on FIRST analysis is given by Wells et al [200].

For further detailed discussion on FIRST, and how to run FIRST see the user guide manual [48] and [67, 82, 83, 84, 109, 122, 200].

Studying protein flexibility is an important and yet complicated biological phenomenon, and FIRST is proving to be a valuable tool in such an undertaking. New refinements and fine tuning of FIRST to better match the biology and experimental evidence are part of ongoing research [122, 200]. In chapter 6 we will address some of these issues and suggest further new extensions.

Protein rigidity and flexibility and its predictions with FIRST will be important in several key chapters (4 - 6). In many cases the rigid cluster decomposition obtained from FIRST will be used as the initial input to our algorithms.

1.4 Thesis Outline

In the initial section of this chapter we have outlined the strategy and the main objectives of this thesis. Specifically, this involves the development of algorithms, models and theoretical verification using combinatorial rigidity. In many cases this will require further refinement and extension of the pebble game algorithm which will allow us to study new problems. We then apply these algorithms and methods to important applications in protein rigidity and flexibility and in mechanical engineering.

We now briefly summarize the rest of the chapters and the key problems and applications that we will address in this thesis.

Chapters 2 and 3 are background chapters, which also includes the review of our previous work, with some new connections and reflections appearing towards the

end of Chapter 3 that is related to the relevant region algorithm. The new work in this thesis appears in Chapters 4, 5, 6 and 7.

- Chapter 2: In this chapter we review the basic concepts in the rigidity theory. We will start by looking at the bar and joint frameworks, and then briefly introduce the body-bar, body-hinge frameworks and the Molecular Theorem as an extension of general Tay's theorem for body-bar frameworks. We will mainly focus on the combinatorial (counting) characterization of rigidity, as it leads to fast pebble game algorithms.
- Chapter 3: In this review chapter we will describe and demonstrate the pebble game algorithm focusing on the $6|V| - 6$ pebble game algorithm as it tracks the critical counting conditions of the Molecular Theorem, which is embedded in program FIRST. We will also summarize several key pebble game properties and invariants that will be important in proofs in Chapters 4 and 5 and 7. We will also describe and illustrate our previous extension of the pebble game algorithm [172], the 'Relevant-regions detection algorithm', and summarize the key properties.
- Chapter 4: In this chapter we will develop a novel protein Hinge-prediction algorithm for predicting hinge locations between rigid protein domains using a single protein structure. This algorithm extends and refines the Relevant-regions detection algorithm to predict the constrained motions between two (or more) rigid clusters (domains) as is the case in protein hinge motions. Some theoretical results relating to this algorithm will also be provided. The Hinge-prediction algorithm will be applied and illustrated on several classic hinge-bending proteins, and other hinge-bending proteins from the biochemistry literature. In addition to the precise and very specific prediction of hinges, for instance non-covalent

hinge locations can be predicted, we can also extract other useful information with this algorithm.

- Chapter 5: In this chapter we will look at the problem of allosteric communication in proteins. Allostery in proteins is an important regulator of many cellular processes, and involves communication between two (or more) distant binding sites, where binding (releasing) of a ligand at one site causes functional important shape changes at a second (distant) site. The exact mechanism of this functionally important signalling through allosteric structural change propagation is elusive and not well understood.

We will show how often very small conformational and/or rigidity changes at one site can propagate through the graph and sample toy model structures and affect the rigidity and shape, often dramatically, at a second distant site. Mathematicians in the rigidity theory have been interested in allostery for some time now.

Motivated by the concept of shape induced changes and transmission over a distance, we will derive several (combinatorial) rigidity-based allostery models, and develop corresponding rigidity-based allostery detection algorithms (using various extensions of the pebble game and relevant-regions detection algorithm). These algorithms are designed to check for rigidity-based (i.e. transmission of rigidity/shape) allosteric communication between two distant sites in the constrained (molecular) multigraph. We will rigorously verify these algorithms, illustrate them on the multigraphs and derive several graph theoretical results on the rigidity-based allostery communication. Once the theoretical concepts and algorithms are introduced, we will apply these algorithms and methods on some proteins, including the important G-protein coupled receptors, which are widely believed to be allosteric.

- Chapter 6: In this chapter we investigate predictions of rigidity/flexibility of protein ensembles using FIRST and hydrogen/deuterium exchange. Often protein structures in the protein data bank are represented as an ensemble of structures (i.e. for instance NMR models). For some time we had considered how to extend and refine the rigidity techniques (pebble game algorithm) and the mathematical model of hydrogen bonds and hydrophobic contacts so FIRST can better handle the ensemble data (when available) rather than relying solely on rigidity/flexibility predictions of the single structure (i.e. snapshot of the protein).

At the BIRS 2008 workshop: ‘Rigidity, Flexibility, and Motion: Theory, Computation and Applications to Biomolecules’, Professor Derek Wilson presented experimental hydrogen-deuterium exchange data (which is an ensemble measure) for a protein Acylphosphatase from hyperthermophile *Sulfolobus solfataricus* (Sso AcP). He asked can rigidity techniques and program FIRST be used to give comparable computational predictions of rigidity. This has resulted in a collaboration with D. Wilson, and this work forms Chapter 6.

In this chapter, we will introduce a FIRST-ensemble method which extends FIRST to give a single rigidity/flexibility prediction using the information from the entire ensemble (i.e. all structures). Moreover, we combine the ensemble solvent accessibility data together with FIRST-ensemble rigidity predictions and propose a novel computational method for predicting hydrogen deuterium exchange. The predictions will be compared with the experimentally determined hydrogen-deuterium exchange profile of Sso AcP.

- Chapter 7: In this chapter, which is joint work with Offer Shai, a mechanical engineer and Walter Whiteley, we will develop combinatorial rigidity techniques and algorithms in important decompositions of linkages (pinned bar and joint

frameworks) in mechanical engineering. More specifically, we will offer several new theoretical characterizations of Assur graph, which are the minimal basic components of linkages. We will also develop a new pinned pebble game algorithm that is played directly on the pinned graph, and when combined with the techniques of directed graphs, it is used to decompose a linkage into Assur graphs. The pinned pebble game algorithm also has other important practical engineering applications in studying the rigidity and motions of linkages.

At the beginning of each chapter we will provide the ‘overview’ section, highlighting the main results and ideas presented in the chapter. For the key chapters 4 - 7, a ‘concluding remarks and future work’ section will also be provided. In addition to the chapter summary, in the concluding sections of each chapter, we will provide an extensive discussion of the future work directions and other research avenues that should be further investigated. We will also outline other possible applications that could be addressed with the methods and algorithms we have developed that were not investigated in this thesis. We chose to put all the concluding remarks and future work considerations at the end of each chapter, rather than in a separate concluding chapter, as the work in each chapter applies and develops theory, algorithms and methods designed for a specific problem and application.

We now turn to the introduction of the rigidity theory.

Chapter 2

Rigidity Theory: Background and Searching for the Counts

2.1 Overview

In this chapter we summarize the basic background of the rigidity theory.

Rigidity comes in several flavours, such as: *finite* or *continuous rigidity*, *infinitesimal rigidity*, *combinatorial rigidity*, *global rigidity*, etc [70, 71, 203, 205]. Each kind of rigidity has its unique purpose and applicability, depending on the intent of study, the type of structure/framework that is studied and the nature of the application. We begin the chapter with an informal discussion of the rigidity of the commonly studied structures, the *bar and joint* structures. Then we will move into the more formal concepts of rigidity, particularly the ‘finite’ (or continuous) rigidity and ‘infinitesimal’ rigidity. We will provide the standard definitions and illustrations, leaving other details which can be found in the references provided. We will then move to the combinatorial and generic characterization of rigidity, which only relies on the topology (underlying graph) of the framework and ignores the possible complications from the geometry of the structure.

Our goal throughout this thesis is to primarily focus on the combinatorial rigidity theory (also referred to as *graph rigidity*) of the structures, as the combinatorial results provide us the key mathematical certificates and tools that often lead to numerous important applications of rigidity theory. Specifically, the combinatorial (graph) characterizations of rigidity (i.e. counting of edges and vertices), lead to fast and efficient *greedy algorithms*, such as the pebble game algorithm. The pebble game algorithm will be stated and illustrated in great depth in Chapter 3 with various important extensions which will also appear in the important applications in Chapters 4 – 7. Unlike the unsolved combinatorial characterization of rigidity for the bar and joint frameworks in 3-space, combinatorial rigidity results are very rich and well developed for the body-bar and molecular structures, which will be our focus. We will outline these results towards the end of this chapter. Combinatorial results and the corresponding pebble game algorithms for these structures are particularly important as they are used to give fast predictions of protein flexibility and rigidity with programs such as FIRST, which are central to our work.

2.2 Rigidity Theory – introduction

2.2.1 Basic concepts

We begin with the most widely studied structures, the ‘bar and joint’ frameworks, which are composed of bars (rods) and joints. In Sections 2.2.2 and 2.3 the more formal definitions are provided. We will primarily illustrate the concepts and definitions using the 2-dimensional structures, as almost all of the discussions and definitions have natural extensions and generalizations in 3-dimensional and higher dimensional space. All discussion is in the Euclidean space.

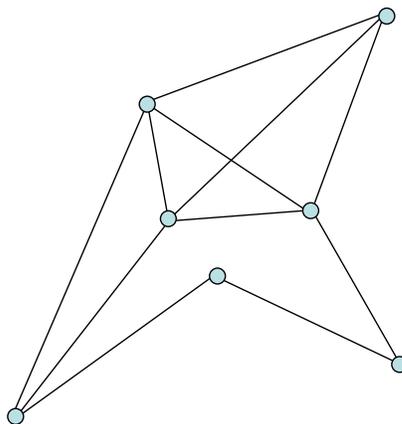


Figure 2.1: A bar and joint framework.

In bar and joint frameworks (or frameworks in short) (Figure 2.1 and 2.2) we assume that the bars (which connect a pair of joints) are strong and rigid (will not get shorter, longer, or break) and the joints, also known as *universal joints* (or *ball joints*) are flexible (freely rotatable) [150]. The joints (points in the 2-dimensional space) basically serve as a connection between a collection of bars, which impose the restriction that the bars share the common endpoints. The bars act as distance constraints, which fix the distance between some pairs of joints. Imagine we want to *move* (continuously) such a framework in the plane, then the distances between pairs of joints that are connected by a bar will remain fixed throughout the motion ¹. The natural question that rigidity theory poses is: will the distances between other (non-connected) pairs of joints also remain fixed?

We are clearly interested in the motions of framework, as its possible motions will guide us to the answers about rigidity/flexibility. Before we provide precise definitions (see next subsection), let us say that a *deformation* (or *flex*) is a motion which preserves the lengths of all the bars of the framework but changes the distance

¹In rigidity theory, there are also other types of structures called tensegrities, where some (or all) bars are replaced with cables (the distance between joints can shrink but not expand) and struts (the distance can expand but not shrink) [205]. We will not be looking at tensegrities in our work.

between some (at least one) pair of (unconnected) joints of the framework. If no deformation exists, then any motion is said to be a *rigid motion* of a framework (i.e. a rigid motion of the framework preserves the distance between all joints in the framework). A framework is *rigid* if it has no deformations (i.e. all of its motions are rigid motions), and is *flexible* otherwise (see Figure 2.2)². Rigid motions (or *rigid body motions*) are often called *trivial motions*, and any motions arising from a deformation are known as *non-trivial motions*. We are clearly only interested in motions of a framework that go beyond the *ever-present* trivial motions (i.e. translations and rotations and their combinations).

One fundamental concept that is central to analysis and description of rigidity/flexibility and motions is the *degrees of freedom* of a framework, or DOF in short. The term degrees of freedom is often encountered in many other fields (in chemistry, engineering, robotics, etc.). Since our work is very interdisciplinary with various applications, and because technical vocabulary and terminology often presents a barrier in communication of ideas among researchers from different fields, we offer several intuitive descriptions of degrees of freedom. The formal mathematical definition is given in the next section.

Roughly speaking, the degrees of freedom is the number of parameters (i.e. independent coordinates or measurements) needed to describe the position of a framework, say in the plane or in three-space [71].

If we consider a single point (joint) moving in the plane, in order to bring this point to any position in the plane, a horizontal and a vertical translation are enough (i.e. translation in the x and y direction). So, we say that a point in the plane has 2 DOF. Another way to think of this is to coordinatize the plane and see that it

²Note that in the rigidity theory literature, there is some ambiguity in the terminology. For instance, flex is sometimes synonymously used for any motion, whether it is trivial or non-trivial [161, 205].

takes two real numbers to identify the location of the point (each coordinate changes independently of the other one). Similarly, in three-space a single point (joint) has 3 DOF.

Now, let us consider two distinct points (joints) P and Q in the plane. P and Q collectively have 4 DOF. Assume we have placed the bar between P and Q (their distance becomes fixed), then we have reduced the total DOF to 3. More specifically, we can bring P to any position in the plane with vertical and horizontal translation (2 DOF). Then we can place Q in any desired position with a rotation about P (1 degree of freedom). So, a single bar (a rigid object) in the plane has three DOF. Moreover, *any rigid framework in the plane with at least two distinct points (joints), has 3 DOF* (combinations of two translations and a rotation)³. Once the position of any two (distinct) points of a rigid framework are fixed⁴ in the plane, the entire framework is fixed (all other points (joints) will be in the fixed positions) [150].

Any bar and joint framework in the plane (with at least two distinct joints) has at least 3 DOF, corresponding to its rigid body motions (generated by 2 translations and 1 rotation). We will often refer to the 3 DOF of a rigid body as the *trivial DOF*. Since trivial DOF are always present in (‘unpinned’ or ‘floating’) framework, it is a useful practice to ignore the 3 trivial DOF and look for any additional DOF corresponding to non-trivial motions. Flexible frameworks have additional DOF, which are called the *internal degrees of freedom* of a framework. Before we give a formal definition, we anticipate that a framework in dimension 2 with more than one joint (point) has n internal DOF, and $n + 3$ total DOF, where n is the number of bars

³It is clear that on the line (1-dimensional space) a rigid body has one DOF (it can only translate along the line).

⁴In the mechanical engineering literature, fixing the position of some joints is called *pinning* or *grounding* of the framework, so the resulting framework, called a linkage, has no trivial rigid body motions. This way analysis can be focused only on the non-trivial motions (if present) of the linkage. Frameworks with no fixed points (‘pinned’ (anchored) joints) are sometimes called ‘floating’, see Chapter 7.

that must be added to rigidify the framework. This is generally a good definition, but we will see shortly that we will need an assumption that joints are not in some special geometry. In order to detect rigidity/flexibility of a framework, it is of clear interest to know how many internal DOF are present. Rigid frameworks have 0 internal DOF.

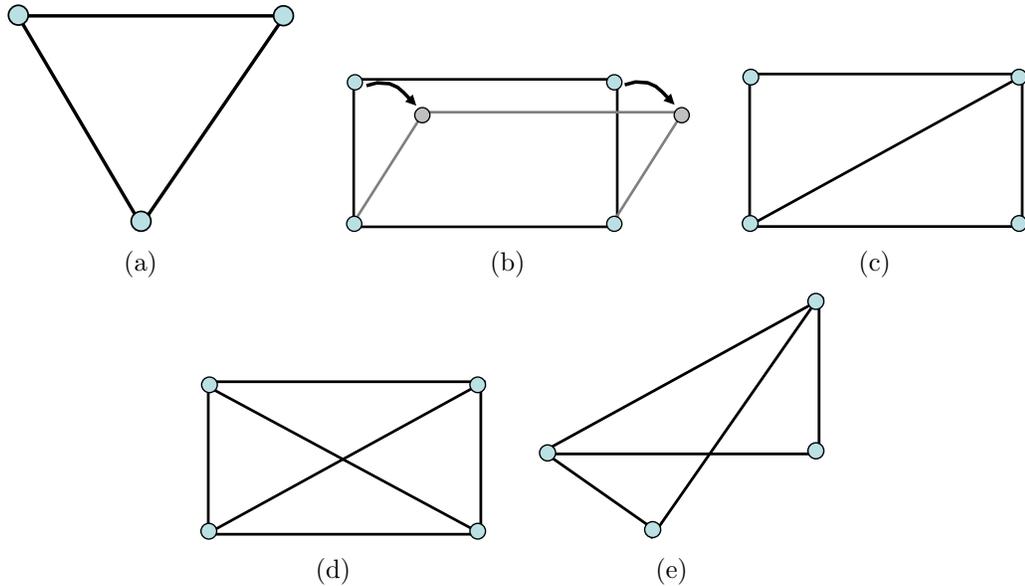


Figure 2.2: Triangle is rigid (a). Rectangle is flexible (b). Rectangle with one diagonal is rigid in 2D, as the distances between all pairs of joints remains fixed (c), additional bars are redundant (d). Motion of 2-dimensional frameworks are restricted to the plane, as clearly rectangle with one diagonal is not rigid in 3D; we can hold one triangle and reflect the other triangle over the diagonal (e).

In Figure 2.2, we have illustrated some simple bar and joint frameworks. Triangle (a) is clearly a rigid framework. Rectangle (b) is flexible, since it deforms into a parallelogram. The rectangle has four total DOF in the plane, the 3 trivial DOF, and 1 internal degree of freedom⁵. The internal DOF corresponds to the extra motion (deformation) when we fix (hold) the bottom two joints, allowing the top and two

⁵Consider the simple counting description of rigidity. If we consider the rectangle (with all four bars removed), it has 8 DOF - five internal DOF. When we place four bars (each bar reduced the degrees of freedom by one), we have removed 4 degrees of freedom, leaving $8 - 4 = 4$ degrees of freedom (1 internal degree of freedom). This intuitive approach works here, but more sophisticated approach and further analysis will be need in matching bars (constraints) with degrees of freedom.

side bars to move. On the other hand, a rectangle with a diagonal (extra bar present) (c) is rigid, it has 3 total DOF (0 internal DOF). Adding another diagonal bar (d) is clearly *redundant* as the framework is already rigid. Frameworks with redundant edges will be called *stressed*. In Figure 2.2 (e) the rectangle with the diagonal obtained by a flip (reflection) of one of the triangles over the diagonal is not congruent to the original framework. Clearly in 3-space, the rectangle with a diagonal is flexible⁶. It is then important that the actual motion of the 2-dimensional framework takes place in the plane. Note also that the framework in Figure 2.1 is flexible (see also Figure 2.7).

Remark. *A rigid framework in 3-space, with at least three non-collinear points (joints) has 6 degrees of freedom.* We give the basic reasoning. For any rigid body in three space, once three of its points (joints) are fixed, the entire body is fixed. Intuitively, three joints collectively have 9 degrees of freedom. The 9 degrees of freedom are reduced to six when the three bars are added forming a triangle (a rigid framework). More specifically, let us denote the three (non-collinear) points in 3-space as P , Q and R . In order to move the point P to any desired position in 3-space takes 3 degrees of freedom (the combination of three translations). Relative to P , we can bring Q to a desired position on a two-dimensional sphere around P . This is a combination of two rotations (think of longitude and latitude on the globe), and it adds two more degrees of freedom. Relative to P and Q , a rotation around PQ axis (1 degree of freedom) can finally bring R to the final position, giving a total of 6 degrees of freedom for any rigid body in 3-dimensional space (see [70, 71, 150] for more useful descriptions). ■

⁶Frameworks that have a single realizations (up to congruence) come up in the subfield of rigidity, called ‘global rigidity’, which is stronger form of rigidity.

2.2.2 Definitions, Rigidity and Infinitesimal Rigidity

All the discussion on rigidity so far was very intuitive and informal. In this section we give precise mathematical definitions and notations. We first state several standard definitions and terms from graph theory.

Graph theory terminology

A *graph* $G = (V, E)$ has a vertex set $V = \{1, 2, \dots, n\}$ and an edge set E , where E is a collection of unordered pairs of *vertices* called the *edges* of the graph. In more extended form, the vertex set and edge set on G may sometimes be denoted as $V(G)$ and $E(G)$, respectively. We say that two vertices v and w are *adjacent* (or neighbours) if edge $e = \{v, w\}$ is present in the graph (i.e. an edge e connects a pair of vertices), otherwise they are *non-adjacent*. The edge $\{v, w\}$ is said to be *incident* to vertices v and w , conversely, the vertices v and w are incident to the edge $\{v, w\}$. To shorten the notation, when no confusion can arise, we may sometimes denote the edge $e = \{v, w\}$ as vw . v, w are called the *endvertices* (or *ends*) of edge vw . The *degree* or *valence* of a vertex v is the number of edges that are incident to v (i.e. the number of edges that have v as an endvertex). A graph G is *complete* if all its vertices are pairwise adjacent (i.e. connected by an edge).

A *subgraph* of $G = (V, E)$ is a graph $G' = (V', E')$, with $V' \subseteq V$ and $E' \subseteq E$, and we simply write $G' \subseteq G$. If $G' \subseteq G$ and G' contains all edges $vw \in E$ with $v, w \in V'$, then G' is called an *induced* subgraph of G . Alternatively, the subgraph induced or *spanned* by a set of vertices is the graph consisting of those vertices and all edges that are only incident to those vertices. The *intersection* of two graphs G_1 and G_2 is a graph $G = G_1 \cap G_2$, where $V(G) = V(G_1) \cap V(G_2)$ and $E(G) = E(G_1) \cap E(G_2)$. The *union* of two graphs G_1 and G_2 is a graph $G = G_1 \cup G_2$, where $V(G) = V(G_1) \cup V(G_2)$ and $E(G) = E(G_1) \cup E(G_2)$.

A path P from (or between) v_0 to v_k is a (nonempty) graph $P = (V, E)$, where $V = v_0, v_1, \dots, v_k$ and $E = v_0v_1, v_1v_2, \dots, v_{k-1}v_k$, where the v_i are all distinct. The vertices v_0 and v_k are called the ends of the path P . The *length* of a path is the number of edges in the path. For brevity, we may denote a path using its natural sequence of vertices (i.e. $P = v_0, v_1, \dots, v_k$). We say that a graph is *connected* if every pair of its vertices is joined by a path. A *cycle* of a graph G is a subset of the edge set of G which forms a path such that the start vertex and end vertex are the same.

In *directed graphs*, edges are ordered pairs of vertices (i.e. edges get a preferred direction, which is usually identified by an arrow on the graph). We say an edge is directed from an *initial vertex* to a *terminal vertex* (see also direction assignment of a graph in Chapter 7 for more detailed definition). The *out-degree* of a given vertex in a directed graph is the number of edges directed out of that vertex.

An edge is *multiple* if there is another edge with same endvertices. A *loop*⁷ is an edge which joins a vertex to itself (i.e. $e = \{v, v\}$). Graphs that do not contain any multiple edges or loops are called *simple* graphs. Multiple edges are permitted in *multigraphs*. The *multiplicity of an edge* is the number of multiple edges sharing the same endvertices. We are abusing the standard notation for multigraphs and identifying a pair of vertices for edges, although technically each edge gets a unique label (see [38]). When there exist multiple directed edges between the same pair of vertices, v and w , such graphs are called *directed multigraphs*.

More details, with illustrations, and other graph-theoretical definitions can be found in any introductory book to Graph Theory (for instance see [38]). ■

⁷In protein structures, a loop is also a common term [15], which is usually an unstructured part of the protein connecting secondary structures (i.e. α -helices, β -sheets) and are not to be confused with the graph theoretical meaning.

We now give more formal definitions of rigidity that were briefly introduced in the previous section. We continue to use 2-dimensional bar and joint frameworks, as definitions easily extend to 3-dimensional (and higher) frameworks.

We define a 2-dimensional *bar and joint framework*, or framework in short, as a triple (V, E, p) , where $G = (V, E)$ is a simple graph (no loops or multiple edges) and a corresponding *configuration* $p : V \rightarrow \mathbb{R}^2$, which assigns each vertex to a point in the plane⁸. The bars are represented by edges and joints by vertices. For simplicity, we will always assume that the endvertices of every edge in the framework have distinct points (i.e. all edge lengths have positive values). Note that in 3-space, the definition is the same, except $p : V \rightarrow \mathbb{R}^3$. For brevity, the framework (V, E, p) is denoted as $G(p)$. As a convenient abuse of notation we denote the point $p(i)$ as p_i , $i \in V$, where in the usual extended form we denote the coordinates of p_i (in 2-dimensions) by (x_i, y_i) . A standard and useful practice (which we adapt) is to identify p as a single point in \mathbb{R}^{2n} ($n = |V|$) (i.e. $p = (p_1, p_2, \dots, p_n)$) [71, 203].

A *motion* (or finite motion) $p(t)$ of the bar and joint framework $G(p)$ is a family of smooth functions $p(t) = (p_1(t), p_2(t), \dots, p_n(t))$ ⁹, $0 \leq t \leq 1$, such that $p(0) = p$ (i.e. $p_i(0) = p_i$ for all i), and

$$|p_i(t) - p_j(t)| = |p_i - p_j| = \text{constant, for all } \{i, j\} \in E \text{ and } 0 \leq t \leq 1 \quad (2.1)$$

(eg. Under the motion $p(t)$, the length (Euclidean) of each edge in the framework is kept fixed [71, 78, 203].)

We say that a motion $p(t)$ of $G(p)$ is a *flex* (also non-trivial motion, deformation) if $|p_i(t) - p_j(t)| \neq |p_i - p_j|$ for all $t > 0$, and some $\{i, j\} \notin E$ (i.e. $p(t)$ is not congruent to $p(0) = p$ for all $t > 0$ – distance between some pair of vertices is

⁸ p is sometimes called the embedding function [71].

⁹In the extended form $p(t) = (p_1(t), p_2(t), \dots, p_n(t)) = (x_1(t), y_1(t), x_2(t), y_2(t), \dots, x_n(t), y_n(t))$.

changed). A framework is *flexible* if it has a flex, and is *rigid* otherwise (has only trivial (rigid body) motions) [203]. More specifically, in a rigid framework $|p_i(t) - p_j(t)| = |p_i - p_j|$ for all $i, j \in V$ and $t > 0$. So, in a rigid framework, every motion $p(t)$ preserves the distances of all pairs of vertices (both adjacent and non-adjacent).

We rewrite the equation (2.1) in terms of the usual dot product,

$$(p_i(t) - p_j(t)) \cdot (p_i(t) - p_j(t)) = c_{ij}, \text{ for all } \{i, j\} \in E, \quad (2.2)$$

where each c_{ij} (constant) is a squared length of edge $\{i, j\}$ (i.e. $c_{ij} = |p_i - p_j|^2$).

In the plane 2.2 gives $|E|$ quadratic equations and $2|V|$ unknowns (imposed by distance constraints (edges) in the framework). Finding a solution to this system of quadratic equations is very difficult even for very small frameworks with few edges (see [71, 150] for more details).

Since rigidity¹⁰ of $G(p)$ is difficult to establish, one approach that engineers and mathematicians have used for centuries is to linearize the problem. More specifically, instead of looking for a flex (deformation) directly, one can simplify the quadratic algebra to a more manageable linear algebra, essentially deriving a system of linear equations by obtaining the first derivatives of (2.2). We outline the basics here.

Thinking of t as time and differentiating edge length constraints from (2.2), dividing by 2 and evaluating at $t = 0$, we get

$$(p_i - p_j) \cdot (p'_i - p'_j) = 0, \text{ for all } \{i, j\} \in E. \quad (2.3)$$

where p'_i ¹¹ represents the unknown *instantaneous* (virtual) velocity of the point p_i .

The set of instantaneous (initial) velocities (one for each vertex) $p' = (p'_1, p'_2, \dots,$

¹⁰Note that rigidity in dimension 1 (i.e. line rigidity) is only a question of connectivity of the graph. The framework $G(p)$ in dimension 1 is rigid if and only if graph G is connected [71].

¹¹ $p'_i(t) = \frac{d}{dt}p_i(t) = (\frac{d}{dt}x_i(t), \frac{d}{dt}y_i(t))$ is a velocity vector of p_i .

p'_n) which satisfies condition (2.3) is called an *infinitesimal motion* of a framework $G(p)$ [71, 203]¹².

We can rewrite equation (2.3) as $(p_i - p_j) \cdot p'_i = (p_i - p_j) \cdot p'_j$, and observe that (2.3) basically says that the initial velocities of the endvertices of any edge have equal projections in the direction of the bar (edge) (see Figure 2.3). In other words, an infinitesimal motion assigns a velocity vector p'_i to each vertex i so that the length of edges (bars) present in the graph is preserved to the first order. Naturally, we want to know if infinitesimal motion also preserves the lengths of non-adjacent (i.e. non-edge) pairs of vertices.

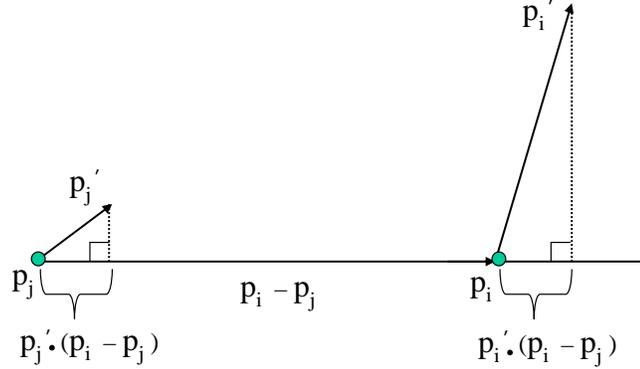


Figure 2.3: *Infinitesimal edge condition.* The length of each edge in the framework stays the same to the first order.

The constraints from (2.3) give $|E|$ linear equations (one per edge) and $2|V|$ unknowns representing p' . Rewriting equation (2.3) as: $(p_i - p_j) \cdot p'_i + (p_j - p_i) \cdot p'_j = 0$, we can write this homogeneous system of linear equations as a single matrix equation

$$R_G(p)p'^T = 0 \tag{2.4}$$

¹²In the literature, infinitesimal motion is sometimes referred to as *first order motion* or *snap-shot motion* [71, 203, 207].

where $R_G(p)$ is called the *rigidity matrix* and $p' = (p'_1, p'_2, \dots, p'_n)$ is a $2n$ -dimensional velocity vector, $p' \in R^{2n}$. The rigidity matrix $R_G(p)$ has $|E|$ rows (one for each edge) and $2|V|$ columns¹³. For each edge $e \in E$, the corresponding row in the matrix has only four nonzero entries corresponding to the difference in the coordinate values of its two associated incident vertices (see below). The rigidity matrix¹⁴ (in any dimension) has this general form [203]:

$$R_G(p) \begin{matrix} & 1 & \dots & i & \dots & j & \dots & n \\ \{i, j\} & \left(\begin{array}{cccccc} \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ 0 & \dots & (p_i - p_j) & \dots & (p_j - p_i) & \dots & 0 \\ \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \end{array} \right) \end{matrix}$$

Consider the graph of a tetrahedron in 2-dimensions (complete graph on four vertices) (Figure 2.4). The rigidity matrix for this framework is:

$$R_G(p) \begin{matrix} & v_1 & v_2 & v_3 & v_4 \\ \{1, 2\} & \left(\begin{array}{cccc} (p_1 - p_2) & (p_2 - p_1) & 0 & 0 \\ (p_1 - p_3) & 0 & (p_3 - p_1) & 0 \\ (p_1 - p_4) & 0 & 0 & (p_4 - p_1) \\ 0 & (p_2 - p_3) & (p_3 - p_2) & 0 \\ 0 & (p_2 - p_4) & 0 & (p_4 - p_2) \\ 0 & 0 & (p_3 - p_4) & (p_4 - p_3) \end{array} \right) \end{matrix}$$

In the extended form (where $p_i = (x_i, y_i)$) this rigidity matrix becomes:

¹³The rigidity matrix of a framework in higher space is almost identical as the one in 2D, with additional columns per each vertex. In 3-dimensions the rigidity matrix is an $|E|$ by $3|V|$ matrix, and in general d -dimensional space it is $|E|$ by $d|V|$.

¹⁴Maxwell in the 19th century was studying the rigidity matrix [125].

$$R_G(p) \begin{pmatrix} v_1^x & v_1^y & v_2^x & v_2^y & v_3^x & v_3^y & v_4^x & v_4^y \\ \{1, 2\} & (x_1 - x_2) & (y_1 - y_2) & (x_2 - x_1) & (y_2 - y_1) & 0 & 0 & 0 & 0 \\ \{1, 3\} & (x_1 - x_3) & (y_1 - y_3) & 0 & 0 & (x_3 - x_1) & (y_3 - y_1) & 0 & 0 \\ \{1, 4\} & (x_1 - x_4) & (y_1 - y_4) & 0 & 0 & 0 & 0 & (x_4 - x_1) & (y_4 - y_1) \\ \{2, 3\} & 0 & 0 & (x_2 - x_3) & (y_2 - y_3) & (x_3 - x_2) & (y_3 - y_2) & 0 & 0 \\ \{2, 4\} & 0 & 0 & (x_2 - x_4) & (y_2 - y_4) & 0 & 0 & (x_4 - x_2) & (y_4 - y_2) \\ \{3, 4\} & 0 & 0 & 0 & 0 & (x_3 - x_4) & (y_3 - y_4) & (x_4 - x_3) & (y_4 - y_3) \end{pmatrix}$$

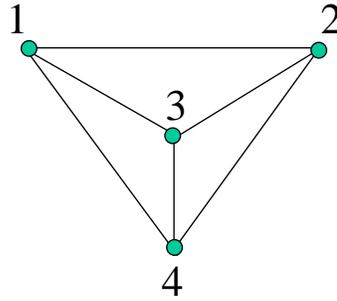


Figure 2.4: Graph of a tetrahedron, a K_4 graph.

Instead of finding the solutions to a system of quadratic equations, we are interested in understanding the linearized, infinitesimal motions (i.e. solution $p' = (p'_1, p'_2, \dots, p'_n)$ of the linear system $R_G(p)p'^T = 0$) of the framework $G(p)$. Infinitesimal rigidity is a natural approximation to rigidity, and their interconnection has been extensively studied (see [70, 71, 150, 184, 203, 205] for details). As we will ultimately rely only on the combinatorial property of rigidity which captures the situations where the two ideas are equivalent (given in next section), we only state the key results along with the standard definitions.

An infinitesimal motion p' is called a *trivial infinitesimal motion* (or *infinitesimal rigid motion*) if it has velocities that arise from some congruence [207] (i.e.

translations and rotations)¹⁵. If the solution p' is not trivial then p' is called an *infinitesimal flex* (or *infinitesimal deformation*)¹⁶. We say that $G(p)$ is *infinitesimally flexible* if it has an infinitesimal flex, and is infinitesimally rigid otherwise. Since infinitesimal (snap-shot) motions of the framework are solutions of the system of homogenous linear equations (solution space), they form a vector space. We are clearly interested in the dimension of this solution space. The space of trivial infinitesimal motions (in the plane) is of dimension three, generated for instance by two translations and a rotation around an origin [71, 203, 207].

A set of edges in a framework is said to be *independent* if their associated rows in the rigidity matrix are linearly independent [78] (i.e. removal of an independent edge decreases the rank by one). An edge is said to be *dependent* or *redundant* if removing it from the framework the rank of the rigidity matrix stays the same. We say that the total *degrees of freedom* of the framework $G(p)$ is the dimension of the solution space of $R_G(p)p'^T = 0$ (i.e. the dimension of the space of infinitesimal motions). The *internal* degrees of freedom of the framework is the dimension of the space of infinitesimal motions minus the dimension of the space of trivial motions (trivial degrees of freedom, i.e. unavoidable solutions)¹⁷. That is, if $G(p)$ has at least two vertices, the internal DOF = total DOF – 3 trivial DOF. An infinitesimally rigid framework $G(p)$ has 0 internal DOF.

Since infinitesimal rigidity can be analyzed using the rigidity matrix (i.e. knowing the size of the solution space), we can make use of the linear algebra tools. A standard result in the linear algebra for a homogeneous system of linear equations

¹⁵Note that if we consider that no two vertices are identical, and no three vertices lie on a line (i.e. vertices are in *general position*) then the infinitesimal motion (in the plane) p' is trivial if $(p_i - p_j) \cdot (p'_i - p'_j) = 0$, for all $i, j \in V$ (see [71, 203, 205]).

¹⁶When only considering general position of vertices then infinitesimal motion is not trivial if $(p_i - p_j) \cdot (p'_i - p'_j) \neq 0$ for some $\{i, j\} \notin E$ (see [71, 203, 205]).

¹⁷In 2-dimensions, a framework with at least two (distinct) vertices always has 3 trivial degrees of freedom (generated by two translations and one rotation about some point).

connects equations, solutions and variables: dimension of the solution space = # columns (variables) – # of independent equations (rank). We can now state a result which says that infinitesimal rigidity of a framework can be determined by simply computing the rank of the rigidity matrix [70, 71, 78, 150, 184, 203, 205]:

Theorem 2.2.1 *A 2-dimensional framework $G(p)$, with $|V| > 1$, is infinitesimally rigid if and only if the rigidity matrix $R_G(p)$ has (maximal) rank = $2|V| - 3$.*

Edges (bars) act as constraints on the possible motions of the framework, and intuitively, this says we need $2|V| - 3$ independent edges to attain infinitesimal (first order) rigidity (note that 3 DOF of a rigid body are not removed).

For completeness, the result for the 3-dimensional frameworks is ¹⁸:

Theorem 2.2.2 *A 3-dimensional framework $G(p)$ with $|V| > 2$, is infinitesimally rigid if and only if the rigidity matrix $R_G(p)$ has (maximal) rank = $3|V| - 6$.*

Unlike the complicated quadratic algebra, we can now simply compute the rank of the rigidity matrix (for instance by Gaussian elimination) and determine if a framework is infinitesimally rigid. Note that any (2-dimensional) infinitesimally rigid framework will have an infinitesimally rigid subframework with exactly $2|V| - 3$ independent edges (simply discard the redundant edges – edges corresponding to rows of zeros after row-reduction).

The natural question is: what is the relationship between the regular rigidity (finite – continuous motions) and the infinitesimal rigidity (‘virtual’ – infinitesimal motion)?

A well known result in rigidity theory, says that if we have regular flex (finite deformations, non-trivial motion) it follows that we have infinitesimal flex [71, 78, 150, 203, 207]:

¹⁸In general d-space it would say: A d-dimensional framework $G(p)$ with $|V| > d$, is infinitesimally rigid if and only if the rigidity matrix $R_G(p)$ has (maximal) rank = $d|V| - \binom{d+1}{2}$ [71, 78, 150, 203, 205].

Theorem 2.2.3 *If a framework $G(p)$ is flexible, then $G(p)$ is infinitesimally flexible. Equivalently (by a contrapositive), if $G(p)$ is infinitesimally rigid then $G(p)$ is rigid.*

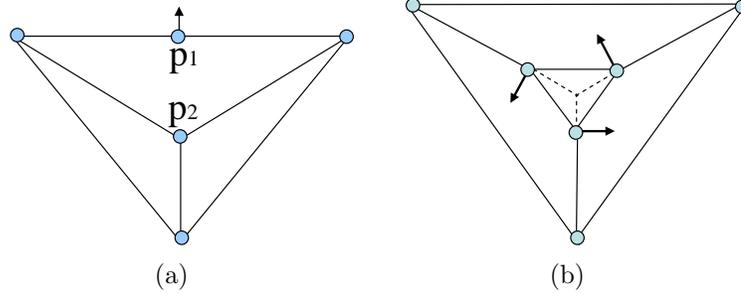


Figure 2.5: Examples of nongeneric (degenerate) frameworks. Both frameworks are rigid but not infinitesimally rigid (i.e. rank is less than maximum rank), the arrows represent non-zero velocities. These cases are extremely rare, and occurs because of the special geometry, top three vertices are on a line (a), three lines from big triangle to small center triangle meet at a centre point (b).

In some special and extremely rare configurations p of the framework, the converse of Theorem 2.2.3 may not be true (i.e. frameworks behave atypically). That is, for some particular p , a framework can be rigid and yet not infinitesimally rigid. For example, the framework in Figure 2.5 (a) is rigid (has only trivial rigid body motions), but it is not infinitesimally rigid (it has an infinitesimal deformation). To see this, we can assign a zero vector to every vertex in the framework, except to vertex p_1 we assign a small non-zero velocity vector (p'_1) (indicated by an arrow) perpendicular to the “chain” (top segment). This (non-trivial) infinitesimal motion (flex) will not distort any edge lengths in the framework at first order, but it will distort the distance (by an infinitesimal amount) say between p_1 and p_2 . One might think of this infinitesimal motion as the vibration of the chain (see [71]). This type of rigidity can be described as “unstable” or “shaky”. Similarly, in Figure 2.5 (b), we can assign zero velocities to the vertices of the large triangle, but the velocities assigned to the vertices of the small triangle correspond to an infinitesimal flex (non-trivial infinitesimal motion) (i.e. hold large triangle and slightly rotate small triangle about

the intersection point of the three lines extending the edges joining the two triangles) [161].

These *degenerate* (or singular) cases like the ones in Figure 2.5 are fortunately very uncommon, and only occur because the framework has special geometry. The special configurations p of a framework that lead to this type of atypical behaviour in rigidity are often referred as degenerate (singular, non-generic) configurations. For instance, in Figure 2.5 (a) this occurs because the top three vertices are collinear). In fact, if we randomly pick the positions of the vertices of the framework (configuration p), it is probability 0 that the framework will be in a degenerate configuration (see [78, 203, 207]). We are interested in the *generic configurations* (not degenerate), which occur for *almost all* configurations (with probability 1 when randomly chosen) [207]. Loosely speaking, a configuration p of a framework is *generic* if we can slightly ‘wiggle’ (i.e. ‘perturb’) the coordinates without altering any of its rigidity properties [161]. This can be explained and defined more precisely again from the rigidity matrix. If we vary the configuration p of a framework (for fixed graph), the rank of the rigidity matrix may change. For instance, if in the framework in Figure 2.5 (a) the three top vertices are slightly perturbed so they are no longer collinear, then the rank of the rigidity matrix will be increased by one (i.e. from 6 to maximum rank = $2(5) - 3 = 7$ on five vertices, and the framework would become infinitesimally rigid by Theorem 2.2.1). We say that the generic configurations p are those configurations which achieve the maximum possible rank of the rigidity matrix on all subgraphs (and configurations that do not give maximum rank are non-generic, singular), details can be found in [71, 203, 207]¹⁹. A framework $G(p)$ with a generic configuration p is often referred to as a *generic framework* (sometimes called regular).

¹⁹Another commonly used (although stronger) definition of generic: a configuration p is generic if all its vertex coordinates are algebraically independent over the rationals. That is to say, any non-zero polynomial with rational coefficients will not vanish when the variables are replaced by the coordinates of the configuration p .

We state this important result due to Gluck(1975) [65] which was also formulated in [6] that connects rigidity with infinitesimal rigidity for generic configurations (almost all configurations) of a framework [203, 207]:

Theorem 2.2.4 (*Gluck*) *For a generic configuration p , a framework $G(p)$ is rigid if and only if $G(p)$ is infinitesimally rigid.*

Equivalently (by the contrapositive), a generic framework $G(p)$ is flexible if and only if $G(p)$ is infinitesimally flexible.

This result ²⁰ assures us that (for generic frameworks) tracking infinitesimal rigidity/flexibility is actually tracking regular rigidity/flexibility (corresponding to finite motions). Note that this result also tells us that if the framework is rigid for one generic configuration (realization) of the framework, it is rigid for any other generic configuration (i.e. almost all configurations) [205]. For further details on infinitesimal rigidity see [71, 150, 203, 205].

Since we are only interested in the generic configurations (ignoring ‘exceptional’ cases), we can confidently discard the prefix “infinitesimal”. In the rest of this chapter and the following chapters we will only be concerned with rigidity/flexibility of the generic frameworks. In the rigidity theory literature it is common to refer to the rigidity of these frameworks as ‘generic rigidity’ or ‘generic flexibility’. With the understanding that we are assuming the framework is generic, in the sequel, as much as possible, we will basically refer to frameworks as rigid or flexible.

²⁰Note that Gluck’s theorem holds for frameworks in any dimension.

2.3 Counting for rigidity in plane graphs and Laman's Theorem

The significance of Gluck's result (Theorem 2.2.4) is that rigidity and flexibility of (generic) frameworks is entirely a property of the graph $G = (V, E)$. In other words, the geometry (i.e. configuration p , edge lengths) of the generic framework $G(p)$ is not important in deciding whether the framework is rigid or flexible. That is to say, *(generic) rigidity is a combinatorial property*. In this spirit, we will step back from our discussion about rigidity/flexibility of frameworks and strictly talk about the rigidity/flexibility of a graph $G = (V, E)$ (graph rigidity can be taken to be synonymous with generic rigidity). It is certainly intriguing to know that rigidity is a property of a graph, but it is not clear how to 'efficiently' check if the graph $G = (V, E)$ is rigid.

Since (generic) rigidity is only a property of the graph, it would be desirable to have a pure graph theoretical (combinatorial) way of deciding rigidity of graphs (i.e. by counting vertices and edges in the graph and its subgraph). We will shortly offer an argument on how to decide (combinatorially) if the graph is rigid. Most of this discussion can be anticipated from Theorem 2.2.1 since the rank of the rigidity matrix offers important clues on what the counting condition for rigidity may look like. The rigidity matrix and rank computation is important as it elucidates rigidity behaviour of the framework and provides us with precise mathematical definitions of rigidity. However, since direct computation of the rank of the rigidity matrix is not practical for our purposes and in applications (i.e. protein rigidity/flexibility predictions), we will now switch the vocabulary and translate some definitions that were stated in terms of the rigidity matrix to pure graph and combinatorial statements to make them more effective. This will match more closely the vocabulary of the graph rigidity literature,

and in rigidity-based algorithms (i.e. pebble game), and will prove to be central as most of our work in this thesis relies on combinatorial rigidity and the pebble game algorithm (see next Chapter).

We say that a graph is *minimally rigid* or *isostatic* if it is rigid and after any one of its edges (but not vertices) is removed, the graph becomes flexible. The main problem in combinatorial rigidity is to find a combinatorial condition with a fast algorithm for the graph to be minimally rigid.

Flexibility in the graph occurs due to unconstrained internal degrees of freedom. Since a single vertex in the plane has 2 DOF, a graph with $|V|$ vertices has at most $2|V|$ DOF (possible independent motions); it has exactly $2|V|$ DOF when no edges are present. Each edge (constraint) eliminates at most a single DOF so in order to eliminate all the internal DOF (recall that 3 trivial DOF are always present), then one could anticipate that at least $2|V| - 3$ ($|V| > 1$) edges are necessary for rigidity of a graph.

Having $2|V| - 3$ edges is clearly not sufficient, as the edges could be crowded among only a select few vertices, leaving the other vertices unconstrained (see Figure 2.6 (a)). The goal then, is to have $2|V| - 3$ *well-distributed* edges. That is, in graphs with $2|V| - 3$ edges, no subgraph on V' vertices should have more than its fair share of $2|V'| - 3$ edges (see Figure 2.6 (b))²¹. If any subgraph $G(V')$ has more than $2|V'| - 3$ edges, then some of these edges are *redundant*. Non-redundant edges are *independent*.

We say that a graph is *stressed* if it has at least one redundant edge. A graph is said to be *independent* if all its edges are independent. A graph is said to be *redundantly rigid* if removal of any edge leaves the graph rigid (Figure 2.6 (c)). Clearly redundantly rigid graphs are stressed (any edge in redundantly rigid

²¹This criterion is sometimes called the *Laman condition*, due to Laman [111], and the graphs (subgraphs) that satisfy this condition are known as *Laman graphs* (subgraphs).

graph is redundant). Redundantly rigid graphs (or subgraphs) are sometimes also called *overconstrained* [79, 94]. Removing a redundant edge from the graph does not change rigidity (i.e. number of DOF in the graph before and after its removal is the same)²². It is the independent edges (i.e. well-distributed edges) that eliminate the degrees of freedom from the graph, so the presence of $2|V| - 3$ well distributed edges should be sufficient for rigidity. This basic intuition is correct in 2-dimensions and it has been stated and used since the time of Maxwell in the 19th century [125]²³. It was finally confirmed in the following famous theorem in rigidity theory, due to Laman (1970) [111]:

Theorem 2.3.1 (*Laman*) *The edges of a graph $G = (V, E)$ are independent in 2-dimensions if and only if no subgraph $G' = (V', E')$ has more than $2|V'| - 3$ edges ($|V'| > 1$).*

Corollary 2.3.2 (*Laman*) *A graph $G = (V, E)$ with $2|V| - 3$ edges ($|V| > 1$) is minimally rigid in 2-dimensions if and only if no subgraph $G' = (V', E')$ has more than $2|V'| - 3$ edges, ($|V'| > 1$).*

Laman’s Theorem is the key graph theoretical result which completely characterizes minimally rigid graphs in 2-dimensional space. Several other equivalent characterizations (see Theorem 2.3.3) have since been discovered [78, 203, 205], and some of these play crucial roles in finding polynomial time algorithms for testing rigidity of graphs, such as the pebble game algorithm, which will be discussed in the next chapter. We will refer to the Laman’s counting condition as *Laman’s count* or a ‘ $2|V| - 3$ count’.

²²Having a series of redundant edges in framework is a familiar idea to engineers who wish to build frameworks (structures) with extra strength and failure tolerance properties [150].

²³Counting for rigidity is also found around the same time in mechanical engineering literature [20, 72].

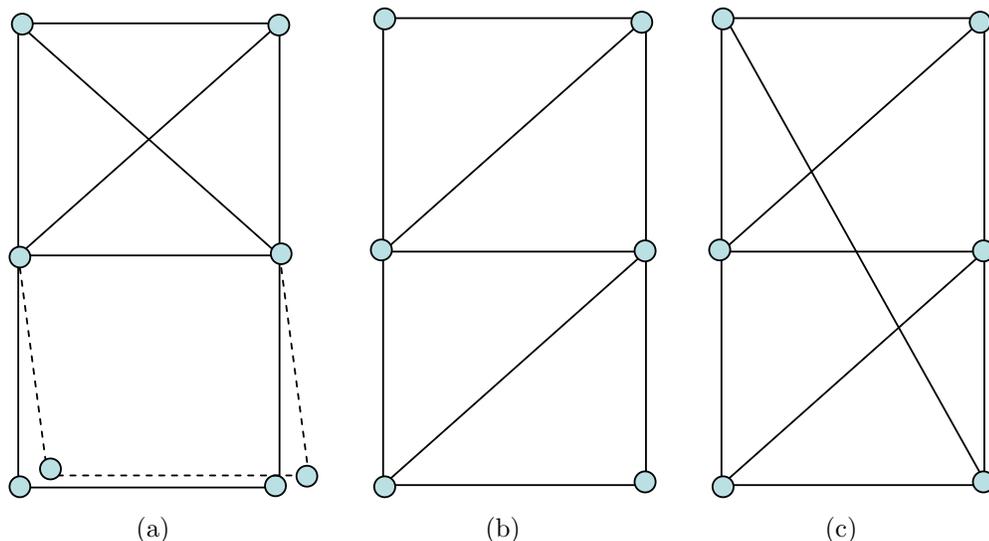


Figure 2.6: Capturing well-distributed (i.e. independent) edge is a key step in characterization of rigidity. Both graphs in (a) and (b) have the required minimum number of edges for rigidity, $2(6) - 3 = 9$ edges. However, in (a) the edges are not well-distributed (independent) so the graph is not rigid. The subgraph induced by the top four vertices has more edges than required (i.e. overconstrained), $2(4) - 3 = 5$ edges are required, but it has 6 edges, which means that one edge is wasted (redundant). In fact, any one of the six edges in that subgraph is redundant (i.e. it is a redundantly rigid subgraph). In (b) all edges are well-distributed (i.e. independent), so the graph is minimally rigid. In (c) the graph is redundantly rigid, removal of any edge leaves the graph rigid.

Let us briefly illustrate the power and elegance of this theorem. In Figure 2.7 we have several different graphs (in 2-space) on same sets of vertices. We apply the Laman's theorem, by simply counting vertices and edges in the graph and its subgraphs. Each graph has seven vertices, so the graph will be minimally rigid if we have 11 ($2(7) - 3 = 11$) edges, and all of the edges are well-distributed (independent). The graph in Figure 2.7 (a) has 11 edges and no subgraph of V' vertices has more than $2|V'| - 3$ edges, guaranteeing its 11 edges are independent. Therefore, from Laman's theorem this graph is minimally rigid (isostatic). In Figure 2.7 (b), we have added an additional edge to the graph, so the graph is rigid (but not minimally rigid). This graph is not redundantly rigid since removal of some edge at a 2-valent vertex makes

the graph flexible. In Figure 2.7 (c) we have an example of a flexible graph. This graph has 11 edges, but one subgraph (indicated by blue edges and its endvertices) has too many edges and does not satisfy the Laman count. All edges in this subgraph are redundant. Finally, in Figure 2.7 (d) the graph is clearly flexible as it has less than 11 edges. These kind of graphs are sometimes referred to as *underconstrained* graphs, as they lack the minimum number of needed edges.

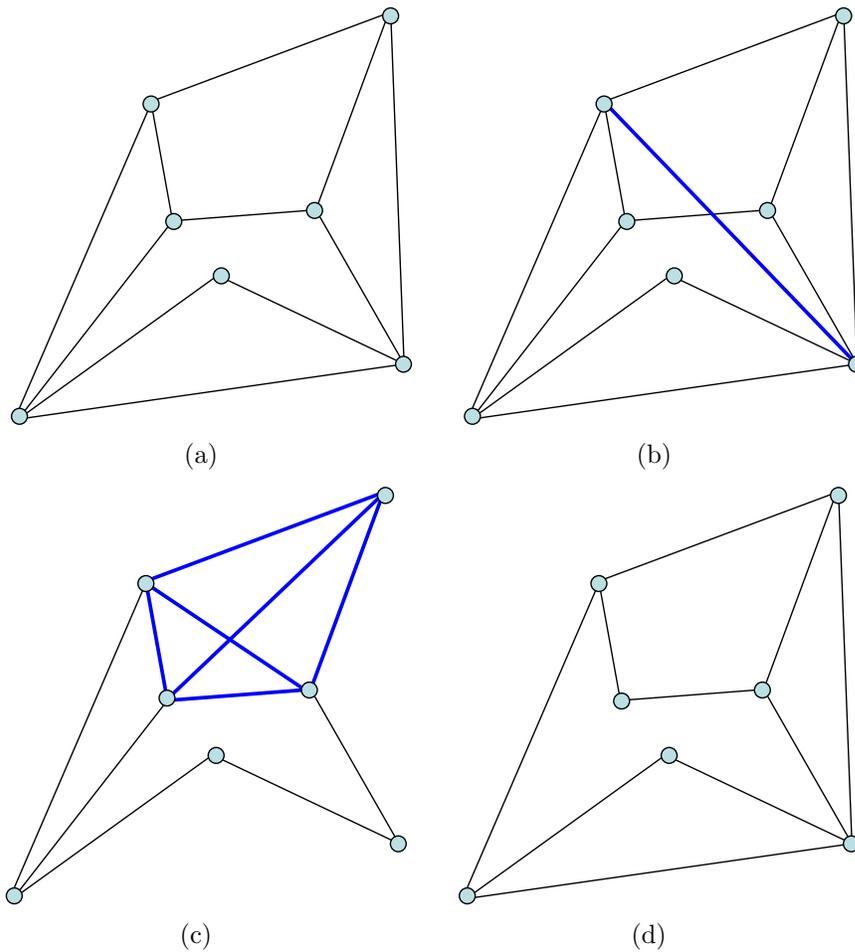


Figure 2.7: Minimally rigid (isostatic) graph (a) as it satisfies the Laman's counts. Extra edge is added (b), so the graph is rigid but not minimally rigid. The graph in (c) has the minimum number of edges, but the blue subgraph violates Laman's count (i.e. has redundant edges), so it is flexible. This graph illustrates that even flexible graphs can have stress (redundant edges), although it is localized in a certain subgraph. In (d) the graph does not have enough edges, so it is flexible.

In Figure 2.8 we have a complete bipartite graph $K_{4,4}$ (see [38] for definition). This graph is rigid, in fact it is redundantly rigid (remove any edge and it remains rigid). By overall $2|V| - 3$ count, it is overconstrained by two edges. This graph illustrates that graphs do not need triangles to be rigid in the plane.

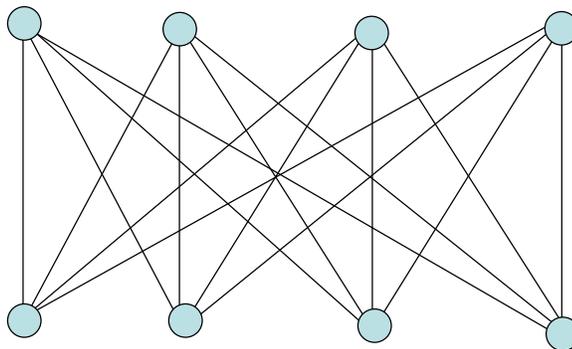


Figure 2.8: A rigid graph with no triangles.

Computing the rank of the rigidity matrix is a significant improvement over the complicated quadratic algebra, but Laman's Theorem is even more powerful. However, in its original form it still leads to a poor algorithm, as it requires counting the number of edges in every subgraph, of which there are an exponential number [78, 79]. The graphs in Figure 2.7 were small enough that we could visually inspect the Laman's $2|V| - 3$ count. The natural next step, which is particularly important for applications, is to find a fast and efficient algorithm that can check the $2|V| - 3$ count in the graph and subgraphs. A standard practice in combinatorial rigidity theory is to look for equivalent characterizations that capture the counting conditions.

As we said earlier, Laman's Theorem has several equivalent formulations. One particular form is [78]:

Theorem 2.3.3 *For a graph $G = (V, E)$ having m edges and n vertices, the following are equivalent.*

- (i) *The edges of G are independent in 2-dimensions.*

(ii) For each edge $\{i, j\}$ in G , the graph formed by adding three additional edges $\{i, j\}$, has no V' vertex subgraph with more than $2|V'|$ edges.

This formulation was initially used by Hendrickson [78], extending the work of Sugihara [180] (as a bipartite matching algorithm) and later revised by Jacobs and Hendrickson [79], and developed into a particularly intuitive and very efficient algorithm, the *pebble game algorithm* ²⁴.

The basic idea behind the pebble game algorithm is to grow a maximal set of independent edges one at a time by matching them to the DOF in the graph (pebbles). A new edge is added if it is determined to be independent of the existing set. If $2|V| - 3$ independent edges are found then the graph is rigid. The pebble game gives a unique and visually appealing way to test edges for independence (track the count), and outputs several key rigidity properties.

In the next chapter we will provide a detailed explanation of the pebble game algorithm for other structures (i.e. body-bar, molecular) that is built in the program FIRST, which have different counts for rigidity. However, the main steps of any pebble game algorithm are very similar and could be adapted for variety classes of counts [114, 115]. In Chapter 7 we will develop a pebble game algorithm for the pinned 2-dimensional bar and joint structures.

2.3.1 Counting is not sufficient in dimension 3

The graph theoretic analysis of rigidity of (generic) bar and joint frameworks in dimension 3 (and higher) is very complicated and not well understood. We have the following necessary counting conditions (in dimension 3) [203, 205]:

²⁴Note that prior to the pebble game algorithm, several other techniques and methods were considered and developed for checking the Laman's counts [180, 203].

Theorem 2.3.4 (*Necessary counts in dimension 3*) *If a graph $G = (V, E)$ is minimally rigid in dimension 3 then it has $3|V| - 6$ ($|V| > 2$) edges and no subgraph $G' = (V', E')$ has more than $3|V'| - 6$ edges ($|V'| > 2$).*

We will call the necessary counting condition in Theorem 2.3.4 a ‘ $3|V| - 6$ count’. Unfortunately, for all of the power that Laman’s counts provides in the plane, the $3|V| - 6$ counting conditions are not sufficient for minimally rigid graphs²⁵. The graph of the “double banana” in Figure 2.9 is the classical counterexample. This graph satisfies the counts; it has the exact required number of edges ($3(8) - 6 = 18$) no subgraph has more than $3|V'| - 6$ edges connecting $|V'|$ vertices. In the context of the previous discussion, all eighteen edges are “well-distributed”, yet the graph is still flexible. Each banana (a rigid subgraph) is made of two tetrahedra sharing a triangle. The two bananas can twist independently around the (imaginary) axis (dotted red line) through their two connecting vertices. This imaginary axis is called the ‘implied hinge’ (or implicit hinge) (see [94, 203] for more details).

It is conjectured that the only reason the $3|V| - 6$ counts are not sufficient in 3-dimension, is because of the presence of implied hinges. This is known as the Dress conjecture [208]. The double banana is the smallest example which shows that the $3|V| - 6$ counting condition is insufficient, however, several other examples have also been generated [123, 203, 205]. In Chapter 7, we will construct similar problematic examples in the pinned bar and joint frameworks.

The issue with the double banana graph, is that its edges are not independent in the sense of Theorem 2.2.2. If we picked some random points for vertices, and constructed a rigidity matrix, we would find that the the rows (edges) of the rigidity matrix are not linearly independent (rank not maximum). Expressing the 3-D independence of edges graph-theoretically is an extremely difficult problem. One of the

²⁵The necessary count in dimension d ($|V| \geq d$) is $|E| = d|V| - \binom{d+1}{2}$ and for every subgraph (V', E') , with $|V'| \geq d$, $|E'| < d|V'| - \binom{d+1}{2}$

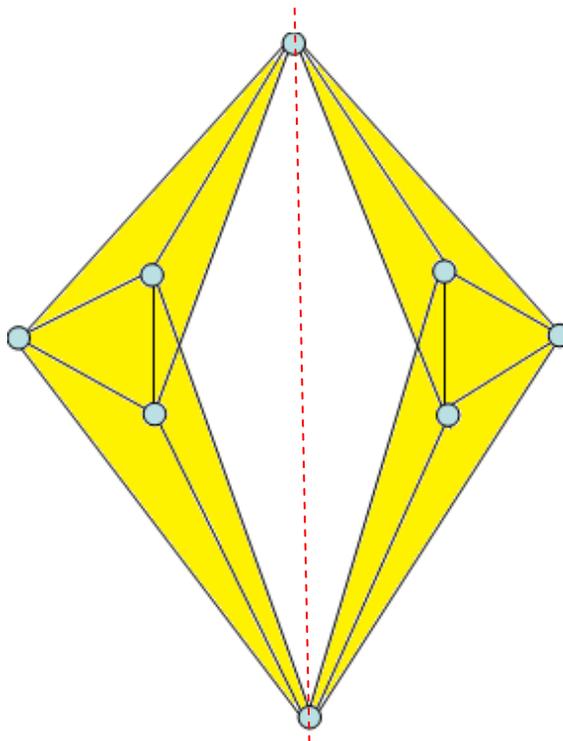


Figure 2.9: A double banana graph shows that Laman type of counts are not sufficient in 3-dimensions. This graph has 18 well-distributed edges, but it is still flexible.

first obvious clues that $3|V| - 6$ counting condition is not sufficient is that it breaks down for a single edge (i.e. when $|V| = 2$, $|E| = 1 > 3(2) - 6 = 0$). Not only do we lack a Laman type of a theorem in dimension 3 for bar and joint graphs, there are currently no known polynomial time algorithms for testing rigidity for general 3-dimensional graphs [207].

Extensive research has been done on this problem and to date we can only trace some partial results [203, 205]. For instance, for some very special class of bar and joint frameworks, related to molecules (see next section) counting results do exist. Finding both necessary and sufficient counting conditions for the rigidity of general bar and joint frameworks in dimension 3 that are both necessary and sufficient remains one of the biggest challenges in the combinatorial rigidity theory. Since many important applications are found in dimension 3, the solution to this

problem would have great practical significance. We now switch our discussion to combinatorial rigidity of different structures which do have necessary and sufficient counting conditions.

2.4 Counting in Body-Bar and Molecular Structures

Why these structures?

So far, we have seen that Laman's Theorem provides both necessary and sufficient counting condition for 2-dimensional bar and joint (generic) graphs, with lack of such counting characterizations in dimension 3 (and higher). We now turn to a different types of structures, that are generally called *body-bar* frameworks (or structures) and more importantly include a special subclass of structures, called *molecular structures*. A surprising result is that these structures have rich and complete combinatorial (counting) characterization (i.e. Laman type of theorems) of minimal (generic) rigidity in dimension 3, with very intuitive and well defined pebble game algorithm (see next chapter). Moreover, these structures have a well developed theory and number of strong conjectures in very difficult area in rigidity – the global rigidity [29]. We should emphasize that we are only interested in the pure combinatorial results which capture the (generic) rigidity behaviour. For background purposes and wider context we will give a brief introduction of the general body-bar and closely related body-hinge frameworks and their combinatorial results (where details can be found in the [203, 204, 207] and several other references provided). However, we are primarily interested in the molecular structures. The nice feature about the molecular structures is that they have an equivalent form in the special class of bar and joint frameworks (bond bending), where the $3|V| - 6$ counts do work on the (generic)

underlying graph [91, 94, 100, 206]. Our primary interest in the molecular rigidity models is because of their uses in applications. In particular, molecular structures are used to mathematically model molecules and together with the corresponding pebble game algorithm on their underlying multigraphs is implemented in program FIRST for protein flexibility predictions. ■

Recall from the earlier discussion in this chapter that a rigid structure (rigid body) with at least three non-collinear points (joints) has 6 degrees of freedom in 3-dimensional space (i.e. trivial rigid body motions). In 3-dimensional bar and joint frameworks, each joint (vertex) is a point with 3 DOF, and then we place a maximum of a single bar (edge) between some pairs of joints, which act as distance constraints. In contrast, in *body-bar frameworks* [203, 204, 207], a body in 3D is a fully dimensional rigid object (i.e. has 6 DOF). Some pairs of bodies are then connected by multiple bars, where each bar preserves the distance between the two *attaching points* (joints)²⁶ which lie on different bodies. The bodies are free to move preserving the distance between any two attaching points (joints) connected by a (rigid) bar. Of course, all joints on a same rigid body move together with the body as a single rigid object and have fixed distances between them. As the bodies move, if they preserve the distance between all pairs of points belonging to different bodies, then the body-bar framework is rigid, otherwise it is flexible.

The body-bar framework may at first be perceived as an unusual structure, although they are very familiar in the robotics and engineering community. Some types of linkages and robotic mechanisms have the structure of a body-bar framework. One famous example is the Stewart platform [179], which is two bodies joined by six bars, which have many important applications (for instance in flight simulations).

²⁶The connection site (which we call attachment points) of the body and one of its bars is taken to be a universal (ball) joint, which has full flexibility [185].

In Figure 2.10 (a) we have shown a body-bar structure, which is composed of just two bodies (rigid objects) that are connected by 5 bars. If we want to join the two bodies into a larger rigid object, then 6 bars between the two bodies are needed ²⁷ [185, 203]. This makes sense, since each body in dimension 3 has 6 DOF, so two freely (independently) moving bodies have a total of $6 + 6 = 12$ DOF and six (properly placed) bars would remove the six non-trivial (internal) DOF (Figure 2.10 (b)), rigidifying the two bodies into a single rigid structure. Since having more than six bars between any two bodies is unnecessary (redundant) and would provide a local overconstraint between the two bodies, for simplicity we can assume that there can be at most 6 bars between any two bodies. If we have a collection of bodies, we would naturally like to know what is the minimum number of bars needed and how should they be distributed to make the resulting structure minimally rigid.

If we think of rigid bodies as vertices and bars connecting the bodies as edges, then the underlying combinatorial structure of the body-bar framework is now a *multigraph* $G = (V, E)$ (with no loops), which allows up to 6 edges between any two vertices. The configuration p of a 3-dimensional body-bar framework $G(p)$ defines the positions of all the attachment points (i.e. end-points) of the bars of $G(p)$ in \mathbb{R}^3 . As we mentioned earlier, we are only interested in the generic behavior of the framework (i.e. use generic points of the attachment points (ends) of each bar), so we will eventually turn to a pure discussion of the rigidity of the multigraph G .

Remark. We should mention that there exist rigidity matrices for body-bar frameworks (which is usually written in the language of Grassman-Cayley projective geometry) (see [91, 203, 204] for details), where rigidity properties with linear algebra tools are captured virtually identically as we described earlier in the chapter for bar and joint structure (for instance, independent edges (bars) correspond to linearly

²⁷It is well known by engineers that six bars are needed to make the two bodies rigid [185].

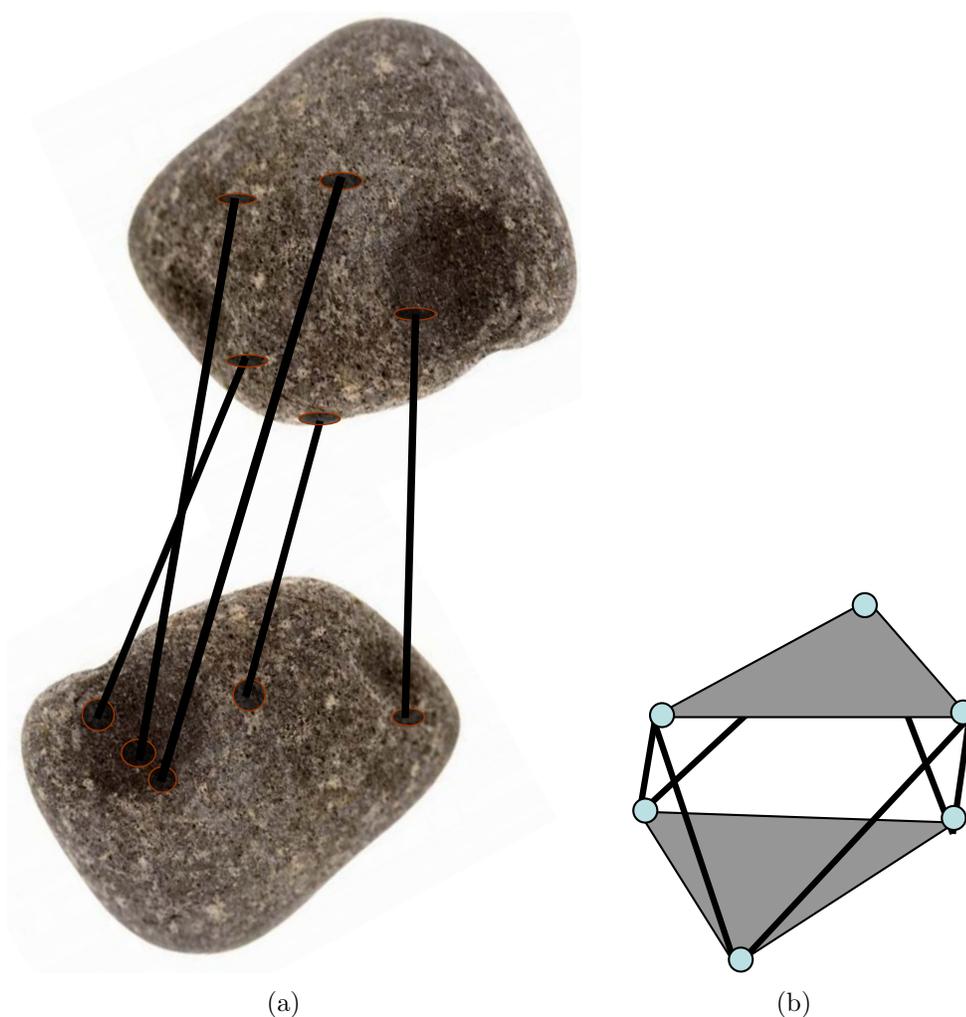


Figure 2.10: Body-bar structures in dimension 3. Two rigid bodies, each with 6 DOF, are connected by 5 bars at the attachment points (universal joints) which preserves the distance between the two attaching points (a). In (b), consider two shaded triangles as two rigid bodies. Connecting two triangles by 6 bars (with generic attachment points) rigidifies (locks) the two bodies into a single rigid structure. Such a rigid body-bar framework can also be viewed as a (generically) rigid bar and joint framework, an octahedron [203].

independent rows in the rigidity matrix). Since we are only going to rely on the combinatorial (counting) results for rigidity (particularly for molecular structures, see below) presenting this here is both unnecessary and distracting. As an alternative, it is possible to model the body-bar framework as a bar and joint framework. To do this

each body is replaced by a (small) isostatic bar and joint structure and together with the bars that are connecting these isostatic frameworks (i.e. bodies), the structure becomes a bar and joint framework (see [29, 130, 203] for details). In doing this the rigidity properties (i.e. DOF) remain the same. As this now becomes a bar and joint framework, the underlying graph is a simple graph (no loops or multiple edges). ■

We now state one of the remarkable results, due to Tay [185], which naturally extends the Laman’s theorem as a both necessary and sufficient counting (i.e. graph property) characterization of rigidity of the (generic) body-bar frameworks.

Theorem 2.4.1 (*Tay’s Theorem*) [185] *A 3-dimensional (generic) body-bar framework $G(p)$ on the multigraph $G = (V, E)$ is minimally rigid (isostatic) if and only if it has $6|V| - 6$ edges and for every nonempty subset $E' \subseteq E$ with vertices V' , $|E'| \leq 6|V'| - 6$.*

We will call Tay’s counting condition a ‘ $6|V| - 6$ count’. As generic behaviour is only a property of the multigraph, we can just talk about the (generic) rigidity of the multigraph G . Tay’s $6|V| - 6$ count completely characterizes the rigidity of multigraph G .

The previous definitions and vocabulary on the bar and joint graphs in dimension 2 using Laman’s $2|V| - 3$ counts can be extended to the (generic) body-bar framework in terms of the Tay’s $6|V| - 6$ counts on the multigraph G . The criterion $|E'| \leq 6|V'| - 6$ in Tay’s theorem ensures that all edges are independent (well-distributed – no subgraph has wasted edges). More specifically, edges in the multigraph G are *independent* if and only if on all the subgraphs on V' vertices they satisfy the count $|E'| \leq 6|V'| - 6$. If some subgraph has more than $6|V'| - 6$ edges, then G will have redundant (dependent) edges. A multigraph G is independent if all its edges are independent. A multigraph G is stressed if it has at least one redundant edge. A multigraph G is redundantly rigid if removing any edge leaves it rigid. A multigraph

G is rigid but not minimally rigid (i.e. it has more than $6|V| - 6$ edges; redundant edge(s)) if and only if there is a subset $|E'| = 6|V| - 6$, and for all non-empty subsets $E'' \subseteq E'$, $|E''| \leq 6|V''| - 6$. That is, a rigid multigraph will have a subset of $|E'| = 6|V| - 6$ independent edges ²⁸.

In graph theory a well known result connects the $6|V| - 6$ counts in the multigraph G with the trees in the graph. A *tree* is a connected graph on a set of vertices with no cycles of edges. A spanning tree of a graph G is a subgraph which is a tree and contains all vertices of G (see [38] for details). The following is a special case of a general theorem of Tutte [191].

Theorem 2.4.2 (*Tutte*) *A multigraph $G = (V, E)$ with $6|V| - 6$ edges, satisfies the count $|E'| \leq 6|V'| - 6$ on all subgraphs if and only if graph G is a union of six edge-disjoint spanning trees.*

Using Tutte's theorem, Tay's Theorem can be restated in terms of spanning trees ²⁹: *A 3-dimensional (generic) body-bar framework $G(p)$ on the multigraph $G = (V, E)$ is minimally rigid if and only if G is a union of six edge-disjoint spanning trees.* The connection of the $6|V| - 6$ count to spanning trees is important as there are many algorithms that can efficiently decompose the graph into edge-disjoint spanning trees [115]. However, in our work we will not rely on the algorithms that directly check for spanning trees.

Standard body-bar frameworks are usually too general for most built structures. One interesting structure, which can be thought of as a special case of the body-bar framework are the *body-hinge* frameworks (or structures) [203, 204]. Again, we just provide the basics.

²⁸We should point out that Tay's Theorem remarkably also extends to body-bar frameworks in higher dimensions, where the general necessary and sufficient count for dimension d is $|E| = \binom{d+1}{2}V - \binom{d+1}{2}$, with subgraph condition $|E'| \leq \binom{d+1}{2}V' - \binom{d+1}{2}$ [185, 203].

²⁹We can write $6|V| - 6 = 6(|V| - 1)$, and see where the six spanning trees come from. Each spanning tree has $|V| - 1$ edges [38]. So, we can think of the $|V| - 1$ as the count of a tree.

In the body-hinge framework, the bodies (vertices), are still fully rigid objects (with 6 DOF), of which some pairs of bodies are connected by a (linear) hinge (lines) [204], where a hinge removes 5 DOF between the two bodies. So for two bodies linked along a hinge, that leaves a total of 7 DOF (6 trivial DOF of a rigid body, and 1 non-trivial (internal) DOF – rotation of the two bodies around the hinge). When the bodies move, they preserve the contacts at hinges (hinges constrain the movement of two bodies in 3-space) (see Figure 2.11 (a)).

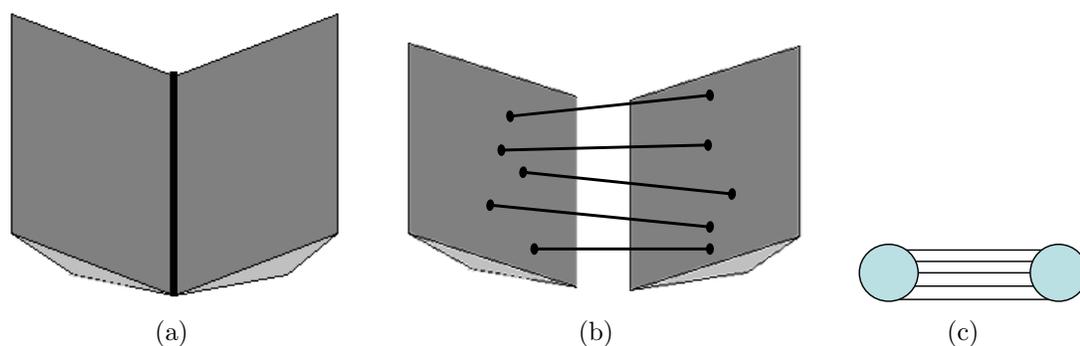


Figure 2.11: A body-hinge framework (a) with the hinge shown as a thick line. The hinge removes 5 DOF leaving 1 ($= 6 + 6 - 5 - 6$ trivial DOF) internal DOF (rotation of two bodies around the hinge). A body-hinge becomes body-bar framework (b) by replacing the hinges with 5 bars (edges). Once hinges are replaced with 5 bars we get a multigraph, with vertices (bodies) and multiple edges (bars) (c). The $6|V| - 6$ counting condition is used to check for rigidity on the multigraph.

The underlying combinatorial structure of the body-hinge frameworks is a (simple) graph $G_H = (V, H)$ (no multiple edges or loops), where V is the set of bodies and H is the set of hinges (edges). We assume that any two bodies can be connected with at most a single hinge (edge), since having two or more (distinct) hinges between the two bodies will just lock the two respective bodies into a single larger rigid unit (see [203]), which should be called a single (rigid) body.

Since each hinge removes 5 DOF, a useful extension is to replace each hinge by a package of five bars (independent edges – each independent edge removes a

single DOF) between the two bodies, so we attain a (special) body-bar structure, and consequently we have transformed the underlying graph G_H into a multigraph (Figure 2.11 (b), (c)). We should note that there are some geometric details in carefully choosing the five bars so they function as a hinge (i.e. the five bars need to pass through the hinge line – axis of non-trivial rotation of one body relative to the second body [203]) but again these details are not important here (see [203, 204, 205]). The most important consequence is that the rigidity of the body-hinge frameworks (i.e. geometrically special body-bar frameworks) can be characterized with Tay’s body-bar $6|V| - 6$ counts. This is confirmed by the following result of Tay and Whiteley [203, 204, 205]:

Theorem 2.4.3 (*Tay and Whiteley - 1984*) *A 3-dimensional (generic) body-hinge framework on a graph $G_H = (V, H)$ is minimally rigid if and only if, each edge (i.e. each hinge) of the graph is replaced by 5 edges, the resulting multigraph contains six edge-disjoint spanning trees on the vertices V .*

If we apply Tutte’s Theorem (Theorem 2.4.2) we can restate this result and see that we are again tracking the counts from the Tay’s Theorem (Theorem 2.4.1). More specifically, once we have replaced each hinge by 5 edges, we can apply the $6|V| - 6$ counting condition on the resulting multigraph to check the underlying rigidity (i.e. independence). So, the combinatorial (counting) simplicity that we have seen in the Laman’s theorem is also applicable using Tay’s counts for body-hinge frameworks.

One important application of the body-hinge frameworks is that they can be specialized to model molecular structures (atoms and their bonds). In the standard molecular modelling kits in chemistry (for example Dreiding model kits, see Figure 5.5 in Chapter 5), the atoms are modelled as rigid bodies and single (rotatable) bonds act as hinges between them. The hinge allows the rotation of one body (atom) relative to the other, about the line through the middle of the bond (hinge) [204]; double or

non-rotatable bonds link the two atoms into a single rigid body and can be treated as such. Geometrically, these are special hinges, as every single bond at an atom meets at the center point of the atom. Roughly speaking, a *molecular structure* (or framework) is a particular case (geometric specialization) of the body-hinge structure where for each body (atom) all the hinges (bonds, lines) at that atom are concurrent in a single point (centre of that atom) [204]. The underlying graph of this structure is a simple graph (no loops or multiple edges) $G_M = (V, H_M)$ where vertices are bodies (atoms) with 6 degrees of freedom and H_M is the set of these molecular hinges (edges) (see [207] for more details).

An astonishing result, which extends the previous results on body-bar and body-hinge structures, says that once the molecular hinges (i.e. single bonds) are replaced by 5 bars (edges), Tay’s $6|V| - 6$ count (and corresponding pebble game algorithm – see next Chapter) completely captures the rigidity of the generic molecular structure. Here, generic is taken to mean that the centers of atoms are generic points. This result is known as the Molecular Theorem, which was originally conjectured more than twenty years ago by Tay and Whiteley [203, 204], and was proven very recently [100]:

Theorem 2.4.4 (Molecular Theorem) *A (generic) molecular structure on a graph $G_M = (V, H_M)$ is rigid if and only if, each molecular hinge H_M is replaced by 5 edges, the resulting multigraph has six edge-disjoint spanning trees on the vertices V ³⁰.*

The Molecular theorem is the key result that is used in applications to protein rigidity and flexibility predictions, and together with the pebble game algorithm (see next Chapter) which tracks the $6|V| - 6$ count in the multigraph, is the main working component of the program FIRST. In FIRST some pairs of vertices (atoms) can have less than 5 edges (bars) (i.e. hydrophobics have 2 – see FIRST manual [48]), up to 6

³⁰Using Tutte’s Theorem 2.4.2, we get again the $6|V| - 6$ counting condition.

edges, where 6 edges lock the two atoms into a single rigid cluster. It is known that even with the modified number of edges to the molecular extracted multigraph, the $6|V| - 6$ counting condition continues to correctly detect rigidity [207, 208].

Tay’s theorem [185] and its various extensions have provided us with the powerful $6|V| - 6$ counting condition, as it gives us a Laman type of rigidity characterization of 3-dimensional structures that have important applications.

From now on, as much as possible, we will no longer refer to any specific subclass (i.e. body-bar, body-hinge, molecular), and we will only talk about the (generic) rigidity of the multigraph $G = (V, E)$, where there can be up to 6 edges between any pair of vertices. However, as we are mainly interested in the applications of the $6|V| - 6$ counts to proteins, the examples that we will investigate in this chapter and subsequent chapters, will be of multigraphs that usually have five edges (i.e. single bonds) between the pair of vertices, but of course this choice makes no difference to the Tay’s counts and the corresponding pebble game algorithm that is given in the next chapter.

In Figure 2.12 (a) we have shown a multigraph on six vertices with five edges between the adjacent vertices. This multigraph has 30 edges, which is the exact minimum number of edges that are needed for rigidity ($|E| = 6(6) - 6 = 30$), and no subgraph has more edges than are necessary (i.e. on all subsets of edges E' , $|E'| \leq 6|V'| - 6$) – all edges are independent (well distributed)). Therefore the multigraph is minimally rigid. This is a graph of a molecular ring of size six (i.e. 6 atoms and 6 single bonds, known as cyclohexane), which is known to be (generically) rigid [204, 207]³¹. Removing any edge from the graph of the ring of six makes it flexible (i.e. it would not have the minimum required $6|V| - 6$ edges), and adding any edge would make it redundantly rigid. See also Figure 2.13. For completeness, we have

³¹Note there are special realizations of the ring of six molecules, (i.e. the ‘boat configuration’), but this occurs because of special geometry (i.e. non-generic) of the positions of atoms [158, 207]

shown that this multigraph (Figure 2.12 (b)) indeed decomposes into 6 edge disjoint spanning trees. We will only discuss the rigidity and the pebble game procedures in terms of the Tay’s $6|V| - 6$ counting conditions and not trees.

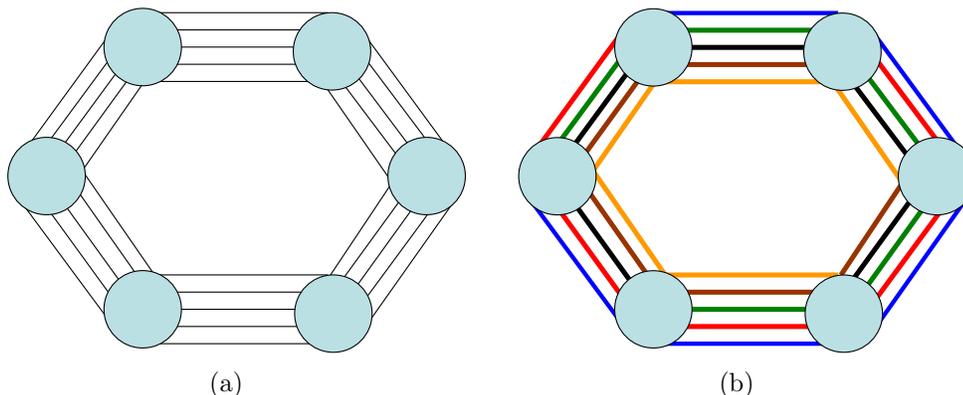


Figure 2.12: A minimally rigid multigraph (a). In (b) we have shown the six edge-disjoint spanning tree (denoted in 6 different colours).

Remark. One very useful property of the molecular structures is that they can be equivalently modelled as a special case of the 3-dimensional bar and joint frameworks. More specifically, we take the molecular graph G_M , and create a *squared graph*, denoted as G_M^2 , which keeps all the vertices (atoms) of G_M and adds an edge to all pairs of vertices of distance exactly two³² (Figure 2.13 (c)). We think of the squared graph G_M^2 as the graph of a (generic) 3-dimensional bar and joint framework (i.e. vertices (atoms) are points with 3 DOF and edges are bars). Note that the next-nearest neighbours (distance 2) are added as they record the constraints that all bond angles between bonds sharing an atom (vertex) are also fixed. It is known [84, 94, 100, 206] that applying the $3|V| - 6$ count on the squared graph captures the same DOF properties as the $6|V| - 6$ count on the multigraph in the Molecular Theorem (i.e. if one is rigid/flexible so is the other one and DOF count is also identical). There is also

³²For a graph G , the squared graph G^2 , has $V(G^2) = V(G)$ with v and w adjacent in G^2 whenever the distance between v and w is at most 2 in G . The distance between two vertices is the shortest path between them [38].

a corresponding $3|V| - 6$ pebble game algorithm that has been described by Jacobs, although it was not mathematically verified [94]. The fact that $3|V| - 6$ counting condition is necessary and sufficient for rigidity of the squared graph is often referred as the *Bar and Joint version of the Molecular Theorem* [91, 94, 100, 206]³³.

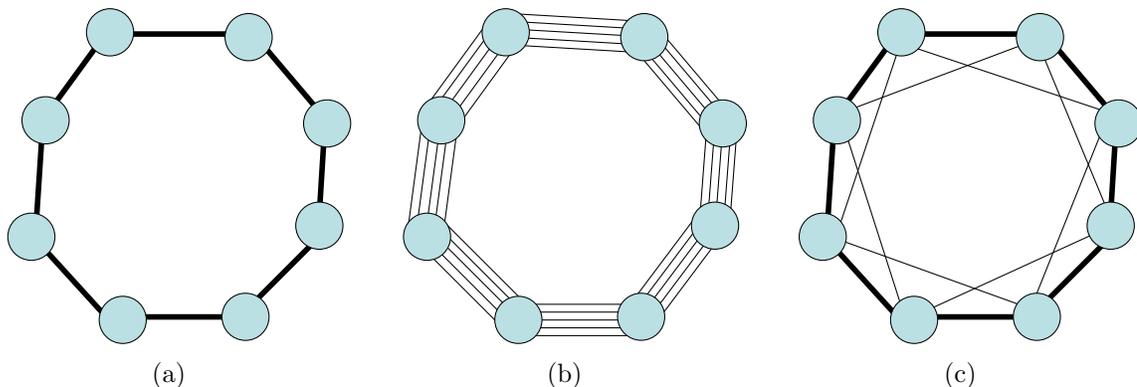


Figure 2.13: Molecule (a) can be modelled and its (generic) rigidity predicted as a body-bar framework where we check the Tay’s $6|V| - 6$ count on the underlying multigraph (5 edges for single bonds) (b) or as a bar and joint framework where we check the $3|V| - 6$ count on the squared graph (c). This is a 8-member ring, and it has 2 internal DOF.

As an example, in Figure 2.13 we have shown a molecular graph G_M of a ring of size 8 (8 atoms and 8 single bonds) (a). Applying Tay’s $6|V| - 6$ count on the multigraph G (i.e. each edge (bond) in G_M is replaced with 5 edges), we see that it is unconstrained by 2 edges, so it has 2 internal DOF (b). More specifically, we need $6(8) - 6 = 42$ independent edges for minimal rigidity, but the multigraph has only $8(5) = 40$ edges. In the G_M^2 graph, it is also underconstrained by 2 edges (c).

³³There are a few small (nonessential) distinctions between the two models, which relate to the one-valent and isolated atoms (vertices), and those need to be removed or handled carefully in the pebble game algorithm, to get equivalence in rigidity [206, 207]. Jacobs [94] also discusses alternate ways of dealing with one-valent and isolated atoms. If detection of redundant rigidity is desired, then small rings of size 3 and 4 is another difference, which come up as redundantly rigid in the body-bar model, and isostatic in the bar and joint model. High-valent atoms also need to be modelled carefully (see [207] for details).

Applying the $3|V| - 6$ count on the squared graph G_M^2 , we need $3(8) - 6 = 18$ edges but it has only 16.

Recalling that necessary $3|V| - 6$ counts (Theorem 2.3.4) are not sufficient for rigidity of general graphs, it is nice to know that on these special class of graphs the $3|V| - 6$ counts are also sufficient. Squared graphs are sometimes referred to as the *bar and joint molecular graphs* or *bond-bending graphs* [94]. ■

As is the case with Laman's $2|V| - 3$ counting condition, directly applying Tay's $6|V| - 6$ counts on the multigraph would lead to a poor algorithm, as we would need to count the number of edges on all subgraphs. We now turn to the pebble game algorithm and its various extensions which can efficiently track these counts.

Chapter 3

The Pebble Game Algorithm and Relevant Regions

3.1 Overview

The pebble game algorithm was initially introduced by Jacobs and Hendrickson [79] as an efficient and visually appealing algorithm that can track the Laman’s $2|V| - 3$ count in the graph (i.e. rigidity of generic bar and joint 2-dimensional frameworks)¹. This extended Hendrickson’s work on bipartite matching and ‘unique realizable graphs’ [78]. Mike Thorpe’s group has extensively applied and revised the pebble game algorithm initially on the flexibility/rigidity predictions of amorphous materials such as glasses (glass-networks) and then to proteins which has resulted in the development of program FIRST [28, 94, 109, 187]. In another thread, Lee, Streinu and others have mathematically verified many useful properties of the pebble game algorithm [114, 115]. In our previous work [172], we have also developed several extensions of the pebble game algorithm and derived and proved a number of important and very

¹Before this paper, a paper by Jacobs and Thorpe already reports first descriptions and applications of the pebble game algorithm [92].

practical properties, which will be restated in this chapter. These extensions and various properties will be summarized as they are central to important protein applications and mechanical linkages that will be presented in the subsequent chapters.

In this chapter, we will describe the pebble game algorithm (sometimes we will just say ‘pebble game’ for brevity), illustrate it on a few examples and highlight important properties. Once we have introduced the pebble game algorithm, we will describe our previous pebble game extension, the *relevant regions* detection algorithm. This algorithm was initially introduced in [172], so this chapter can be regarded as a background and review chapter. We will lay out a number of important properties related to this algorithm. The relevant regions detection algorithm, in various forms, will be used in Chapters 4 and 5, in hinge prediction in proteins and in detection of rigidity based allostery in sampled multigraphs and in proteins.

3.2 The Pebble Game Algorithm

Assume we are given a multigraph $G = (V, E)$ where edge multiplicity is at most six. In this section, we will describe in detail the $6|V| - 6$ pebble game algorithm which tracks the independence of edges and rigidity in the multigraph G . Almost identical pebble game algorithms² can be described for Lamans’s $2|V| - 3$ counting condition (see also Chapter 7 for pinned version of this game) and for many larger classes of counts. The main steps of most pebble game algorithms are very similar. One exception is the $3|V| - 6$ pebble game algorithm that is applied on a squared graph³ (i.e. molecular bar and joint graph). This pebble game is slightly different as it has edge testing priority rules that are not needed for other pebble games with

²When we refer to a different pebble game, we mean a pebble game that tracks a different count on the graph, and we will refer to that pebble game with the corresponding count.

³On simple graphs in dimension 3 (i.e. bar and joint) there is usually no sense in applying the $3|V| - 6$ pebble game as the counts are not sufficient for rigidity (see double banana example in Figure 2.9 in the previous chapter).

more complicated acceptance criteria [28, 94] – see below on meaning of terms. We will not discuss the $3|V| - 6$ pebble game as it is not needed for our purposes (details can be found in [95]). Moreover, this pebble game has not been thoroughly verified and in most implementations to molecules has been replaced by the simpler $6|V| - 6$ pebble game on the multigraph [84].

We should note that $6|V| - 6$ count is a special case of a larger family of $k|V| - k$ counts (k any positive integer), where the pebble game algorithms easily generalize for any k [114]⁴. We are going to strictly concentrate on the $6|V| - 6$ pebble game algorithm because of its application to molecules and protein flexibility predictions. The description of the $6|V| - 6$ pebble game algorithm that is given in this chapter is basically how it is used in FIRST to predict rigidity and various other properties of large multigraphs extracted from the protein structure (i.e. pdb file).

Some of the basics rigidity properties that the pebble game algorithm can determine (which will be shortly described) are [115]:

- (i) declare if the multigraph G is rigid or flexible by determining how many DOF the multigraph has (i.e. number of free pebbles remaining);
- (ii) if the graph is flexible (i.e. has more than 6 total DOF – more than 6 free pebbles) then all maximal rigid subgraphs⁵, also called *rigid clusters*, are determined; these are regions with V' vertices and $6|V'| - 6$ independent (i.e. pebbled – well-distributed) edges;
- (iii) all maximally redundantly rigid subgraphs are determined (i.e. subgraphs which are rigid even if any one edge in the subgraph is removed);

⁴If $k = 3$, we get a $3|V| - 3$ count of 2-dimensional body bar frameworks [203].

⁵It is maximal in the sense that a subgraph cannot be enlarged to another rigid subgraph by including additional vertices and edges.

There are numerous other extensions and properties that can be obtained from the pebble game, which will be discussed later in the chapter.

There are a significant number of terms and rules that need to be clearly explained and defined, which are typically either missing and not defined in the literature or just assumed to be understood in the folklore. Once the terms and the pebble game rules are clearly defined, we will provide the pseudocode and illustrate the pebble game on some sample graphs. Illustrations of this visually appealing algorithm are very important, as they will clarify and sharpen our understanding of the important pebble game rules and its terminology.

The main idea of the $6|V| - 6$ pebble game is to apply Tay's Theorem (in other words the $6|V| - 6$ counting condition) recursively by building the multigraph one edge at a time and growing it to a maximal independent (well-distributed) subset of edges (i.e. maximal set of edges $|E'| \subseteq E$, where $|E''| \leq 6|V''| - 6$ on all subsets $E'' \subseteq E'$). A new edge is added to the independent set (i.e. pebbled) if it is independent of the existing independent set (i.e. already pebbled edges, see below). Once all edges are tested, if we find exactly $6|V| - 6$ independent edges, then the multigraph is rigid (i.e. it has only 6 trivial DOF), otherwise we found less than $6|V| - 6$ independent edges and the multigraph is flexible. By subtracting the total independent edges from $6|V|$ (total DOF with no edge in the multigraph) we can determine how many remaining DOF are in the multigraph.

Recall that to check the $6|V| - 6$ counting condition (i.e. check for independence) we would need to count edges and vertices on all the subgraphs (an exponential number [150]). This is where the pebble game comes in very handy, as the test for independence of an edge in the pebble game is performed in very effective and elegant manner.

The initial step in the pebble game algorithm is to assign ‘pebbles’ to each vertex. The pebbles represent degrees of freedom (DOF) associated with a vertex, where each pebble corresponds to one DOF. In the $6|V| - 6$ pebble game we will assign six pebbles to each vertex as each vertex (body) is a fully rigid body with 6 DOF, so we have a total of $6|V|$ free pebbles. The pebble game proceeds by *placing* (moving) the pebbles from vertices onto incident edges in the multigraph (i.e. edges remove DOF – pebbles). When we *cover* (or pebble) an edge by a pebble, that edge is declared as independent. The key rule of the $6|V| - 6$ pebble game which declares an edge as independent is the following *pebble acceptance criteria*:

- *we can only place a pebble onto an edge (i.e. declare it independent) if the ends of that edge have at least seven combined free pebbles (prior to the placement of the pebble).*

This test for independence of an edge (i.e. pebble acceptance criteria) is one of the most important steps in the pebble game, and the verification that pebbled edges are independent can be found in [115]. More specifically in [115] Lee and Streinu have shown that an edge in the multigraph G is independent if and only if at least seven free pebbles can be gathered on its endvertices before that edge is pebbled. Equivalently (by a contrapositive) an edge is redundant if and only if we are not able to collect seven free pebbles on its endvertices. So, the pebble acceptance criteria basically ensures we will not violate the $6|V| - 6$ count on any subgraph (see below for more comments). When we cover an edge, say $e = \{i, j\}$ by a pebble from vertex i , that edge becomes directed from i to j . So, the pebble game is constructing a *directed multigraph*. Once we get through the rules and terminology, all these ideas will become clear when we illustrate the pebble game on examples (Figures 3.1 - 3.5).

Remark. While it is correct to say that we are moving (placing) pebbles from vertices onto edges (as edges are removing DOF – constraining the possible motions),

a common source of confusion is to think that pebbles are disappearing. The total number of pebbles in the multigraph always stays constant (i.e. $6|V|$ pebbles). To be precise, some pebbles are used to cover edges, and other pebbles will be on the vertices. We will call the pebbles on vertices *free pebbles*. We say that a pebble is *associated* with a vertex v in G if the pebble is either a free pebble on v or if it is covering an edge incident to v . Each vertex will always retain its six associated pebbles as we play the pebble game (see Lemma 3.2.3 property 3 for a nice related property). ■

It quickly becomes apparent as we continue to cover more edges with pebbles (i.e. remove free pebbles – add independent edges) we will not always have seven available free pebbles on the ends of the edge that we are testing. To introduce how the pebble game is still able to cover (pebble) those independent edges, we need to introduce two key pebble game operations: a *swap* and a *cascade*.

Suppose that a given edge $e = \{v, w\}$ is covered by a pebble from v (we say e is directed from v to w). If w has at least one free pebble, then we can return the pebble that is on e onto v and place a pebble from w on e . This move we call a *swap*. After the swap is performed, v has obtained one extra free pebble, and w has one less free pebble. The edge e is still covered by a pebble but its orientation is reversed, as it is now directed from w to v . With the swap operation if we do not have seven pebbles (Figure 3.1 (h)) on the ends of the edge we are testing, we can attempt to perform a swap with adjacent vertices and find more free pebbles (Figure 3.2). If none of the adjacent vertices have free pebbles, we can search over the partially constructed directed multigraph (say with a breadth search), and if the free pebble is found on some vertex (which can be far from v or w) we can perform a sequence of such swap moves until we obtain a free pebble on the ends of the edge e .

We formalize these important pebble operations. Let $P = \{v_1, \dots, v_n\}$ be a directed path constructed by the play of the pebble game (i.e. each edge $\{v_i, v_{i+1}\}$ of P is covered by a pebble, and directed from v_i to v_{i+1} , $i = 1, \dots, n - 1$). Assume v_n has at least one free pebble. We perform a swap between v_n and v_{n-1} , returning the pebble from edge $\{v_{n-1}, v_n\}$ onto vertex v_{n-1} and cover this edge by the free pebble from v_n (i.e. now this edge is directed from v_n to v_{n-1}). We continue to perform a sequence of swaps until an extra free pebble appears on v_1 , and in process we have reversed the directed path P . A sequence of swaps is called a *cascade*. So, under a cascade v_1 has gained a free pebble, and v_n has lost a free pebble. We will sometimes say that we are *recovering* or *drawing* a pebble from v_n to v_1 . However, it is important to note that a common illusion is that the pebbles are being moved from one end of the path to the other. Since all six pebbles associated with any vertex are either free pebbles or covering incident edges, we are clearly never moving any pebbles away from its vertex. With swaps and cascade we are changing the distribution (i.e. location) of free pebbles, and of course keeping all covered (pebbled) edges covered, just changing their orientation. In any pebble game algorithm, once an edge becomes covered (declared independent) it always remains covered.

Using the acceptance criteria and a swap or sequence of swaps (a cascade) we have all the rules to play the pebble game on the multigraph G . This is basically how any pebble game algorithm is played, except the acceptance criteria may change, depending on the underlying count. Using a swap and cascade we can recover free pebbles to the ends of the edge we are testing. If we are not able to draw back seven free pebbles (six free pebbles can always be recovered, see Lemma 3.2.3) on the ends of the edge, that edge is declared redundant and is not covered by a pebble (i.e. it does not remove any DOF). All of this is nicely visually illustrated in Figures 3.1 – 3.3 and Figures 3.4 and 3.5). We now provide the simple intuitive pseudo code

of the $6|V| - 6$ pebble game algorithm. The details that can improve computational performance can be found in [84, 114].

Algorithm 3.2.1 *The $6|V| - 6$ Pebble Game Algorithm:*

Input: A multigraph $G = (V, E)$ (no loops, edge multiplicity at most six).

Initialize $I(G)$ and $\mathfrak{R}(G)$ to an empty set of edges. Place six pebbles on each vertex of G .

Test the edges of E in an arbitrary order.

1. *Until every edge in the multigraph G has been tested, take any untested edge e , and go to step 2. Otherwise go to step 3.*
2. *Count the number of free pebbles on the endvertices of e , say vertex u and v .*
 - (a) *If the vertices u and v have at least seven free pebbles, then place any pebble from either u or v onto e , directing the edge e from that vertex. Place e into $I(G)$ (edge is independent) and return to step 1.*
 - (b) *Else, search for a free pebble from u and v , by following the directed edges (covered edges) in the partially constructed directed graph $I(G)$.*
 - (i) *If the free pebble is found (not considering the pebbles on vertices u and v) on some vertex w at the end of the directed path P (which starts at u or v), we perform a swap or sequence of swaps (cascade), reversing the entire path P , until a free pebble appears on the initial vertex (u or v) of the path P (i.e. w loses one free pebble, and u or v gains one free pebble). Return to step 2.*
 - (ii) *Else, we could not find the seventh free pebble, and the edge is declared redundant (could not be covered by the pebble). Place e into $\mathfrak{R}(G)$ (redundant edges). Return to step 2.*

3. Once all edges have been tested, stop.

Output the sets: $I(G)$ and $\mathfrak{R}(G) = E - I(G)$.

When the algorithm is finished, $I(G)$ is the maximal independent set of edges (edges that are covered by pebbles). $\mathfrak{R}(G)$ is the set of redundant edges (edges that were not covered by a pebble). Total DOF in a graph = number of remaining free pebbles.

At the end of the pebble game there will always be six remaining free pebbles in the multigraph (indicating the 6 trivial DOF – rigid body motions). We started with $6|V|$ free pebbles (prior to covering (pebbling) of any edge), and each independent edge used up one free pebble (DOF). So, once all edges have been tested, the total remaining DOF in the graph is $6|V| - |I(G)| =$ total remaining free pebbles. To get the internal (non-trivial) DOF we take the total remaining free pebbles and subtract six (trivial DOF), that is internal DOF = $6|V| - |I(G)| - 6$. Since pebbled (covered) edges become directed, the multigraph on $I(G)$ is directed. Note that at every stage of the algorithm, the pebble game algorithm is doing nothing more than elegantly keeping track of the Tay’s $6|V| - 6$ counting condition on all subgraphs. It is building the maximal subset of independent edges: at every stage the covered edges satisfy the count $|E'| \leq 6|V'| - 6$ [115] (see Lemma 3.2.3). The initial placement of the six pebbles come from the $6|V|$ portion of the count, and the acceptance condition that at least seven pebbles are present before the edge can be covered (i.e. declared independent - step 2 (a) Algorithm 3.2.1) by a pebble ensures that the critical counting condition $|E'| \leq 6|V'| - 6$ (Tay’s Theorem 2.4.1) is maintained on all subsets of pebbled edges (see also [115, 207]).

This theorem nicely captures the equivalence relations of the output of the pebble game, counts, trees and rigidity:

Theorem 3.2.2 [115] *Let $G = (V, E)$ be a multigraph (no loops, edge-multiplicity at most six) which has exactly $6|V| - 6$ edges. The following characterizations are equivalent:*

- (1) *G is minimally rigid (isostatic).*
- (2) *(Subset or Count) Every subset of $|V'|$ vertices spans at most $6|V'| - 6$ edges (i.e. for every nonempty subset $E' \subseteq E$ with vertices V' , $|E'| \leq 6|V'| - 6$)*
- (3) *(Spanning Trees) G can be decomposed into exactly six edge-disjoint spanning trees.*
- (4) *(Pebble Game) The $6|V| - 6$ pebble game (Algorithm 3.2.1) ends with all edges E covered by pebbles and exactly six free pebbles remaining on the vertices of G .*

The equivalence of (1) and (2) comes from Tay's Theorem (Theorem 2.4.1), equivalence of (2) and (3) is Tutte's Theorem (Theorem 2.4.2) and equivalence of (3) and (4) or (2) and (4) is given by Lee and Streinu [114, 115].

We will shortly provide more observations and important properties, but to make these ideas more clear, it is in order to illustrate and run this algorithm on some examples.

3.2.1 Pebble game illustration and properties

In Figures 3.1, 3.2 and 3.3 we show the output of the pebble game (Algorithm 3.2.1) on the multigraph of a ring of 6, which is minimally rigid as it satisfies the $6|V| - 6$ counts (see also Figure 2.12). It has $6(6) - 6 = 30$ edges and on all subgraphs the $|E''| \leq 6|V''| - 6$ is satisfied (i.e. all edges are independent). Once we have placed six pebbles on each vertex, we pick any edge (red edge in Figure 3.1 (c)) and as long as the ends have at least seven free pebbles (pebble acceptance criteria) we cover the

edge using any free pebble from either end-vertex (Figure 3.1 (a - d) – step 2 (a) of the algorithm). We continue to place pebbles on edges one by one ensuring we always satisfy the pebble acceptance criteria (Figure 3.1 (e - g) - steps 1 and 2 (a) of the algorithm). Each time we place a pebble onto an edge, we direct the edge (indicated by an arrow) from the vertex of the free pebble. When we do not have seven pebbles on the ends (Figure 3.1 (h)) we search over a directed path and if a free pebble at an adjacent vertex is found we perform a swap (Figure 3.2 (i - k) - steps 2 (b) (i)). Sometimes we may have to search out further in the graph and perform a sequence of swaps (a cascade) to recover a free pebble (Figure 3.2 and 3.3 (l - u)). The game stops once all edges have been tested.

Note how reversal of pebbles still keeps all pebbled edges covered by pebbles. However, after the swap(s) the orientation of the directed paths (highlighted in turquoise) is reversed. In this example, all edges are pebbled (declared independent) and there are only six remaining free pebbles (corresponding to six trivial DOF of a rigid body) indicating that the multigraph is rigid. Moreover, as it has no redundant edges (all edges are covered by a pebble), it is minimally rigid.

In Figure 3.4 all edges so far were successfully covered by a pebble. We are testing the red edge, and we notice its ends (yellow vertices) have less than seven free pebbles. We search along the available directed path (b) (in turquoise) and we are not able to find a seventh free pebble. So the pebble search for the seventh pebble is not successful and we are not able to place a pebble onto this edge. This indicates that the edge is not independent (i.e. it is redundant) (step 2 (ii) of the algorithm). We distinguish a redundant edge by a dashed line (Figure 3.5). When the search for the seventh pebble is not successful, we say we encounter a *failed search* [79, 84, 109, 172, 207]. Note that the remaining graph has additional free pebbles but these are not accessible as no directed path from the yellow vertices can direct us to

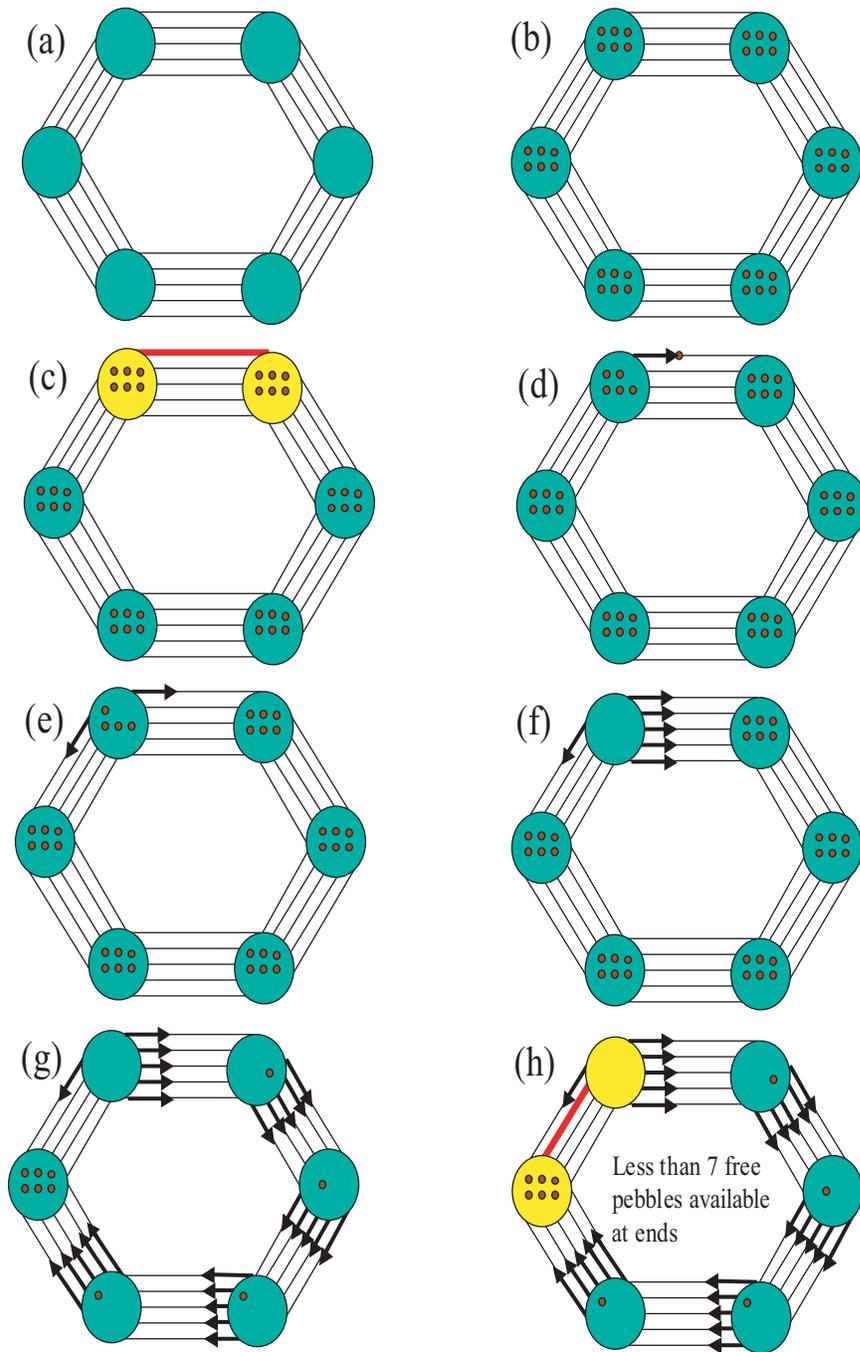


Figure 3.1: *Illustration of the $6|V| - 6$ pebble game algorithm.* 6 pebbles are placed on each vertex. If there is at least 7 free pebbles on the ends of an edge, an edge is covered by a pebble (declared independent). The edges become directed (shown by an arrow). As more edges (constraints) are covered by pebbles, the number of free pebbles (DOF) starts to decrease. See text for more details. Continued in Figure 3.2.

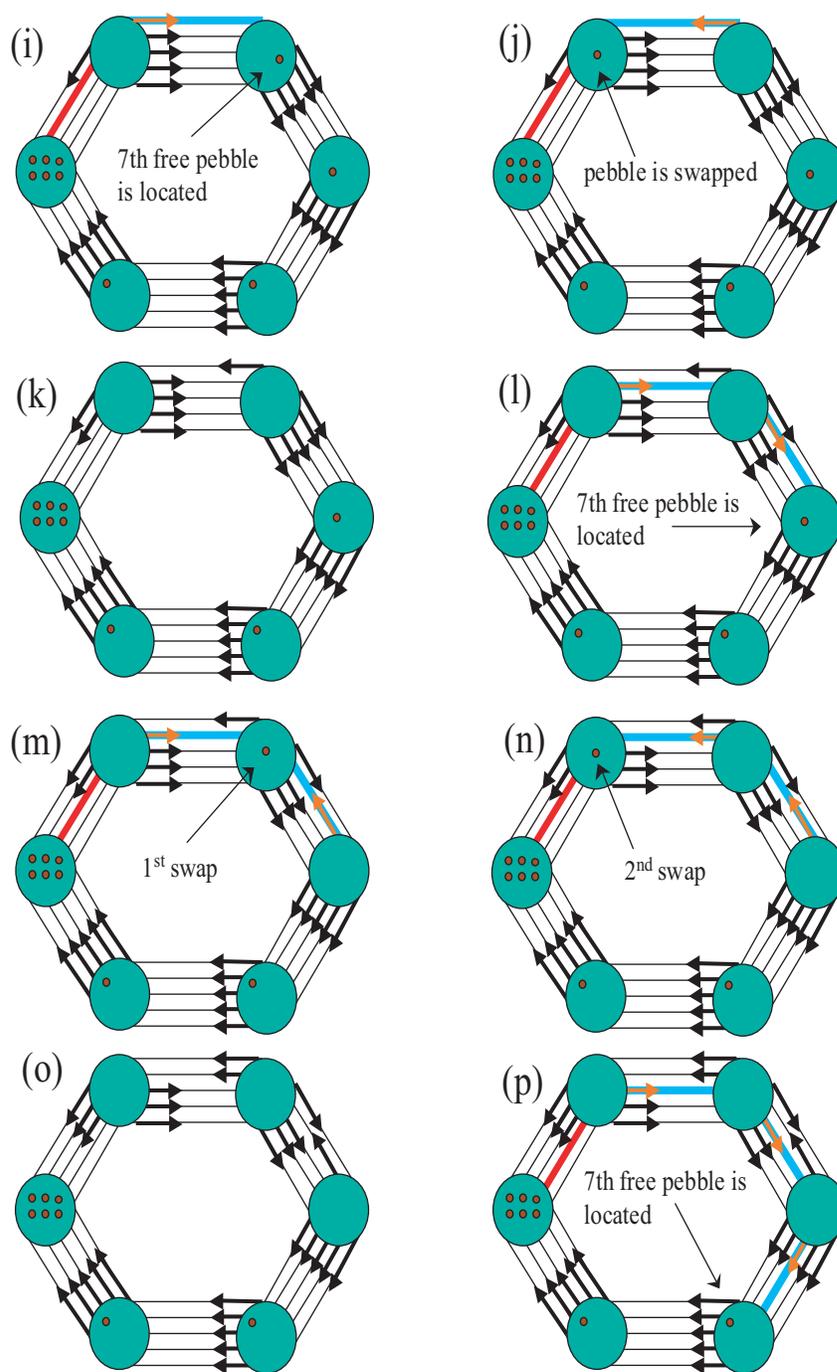


Figure 3.2: *Illustration of the $6|V| - 6$ pebble game algorithm ... continued.* If 7 pebbles are not available on the ends of an edge, we can search in the partially constructed directed multigraph, and reverse free pebbles using a swap or a sequence of swaps (a cascade). Continued in Figure 3.3.

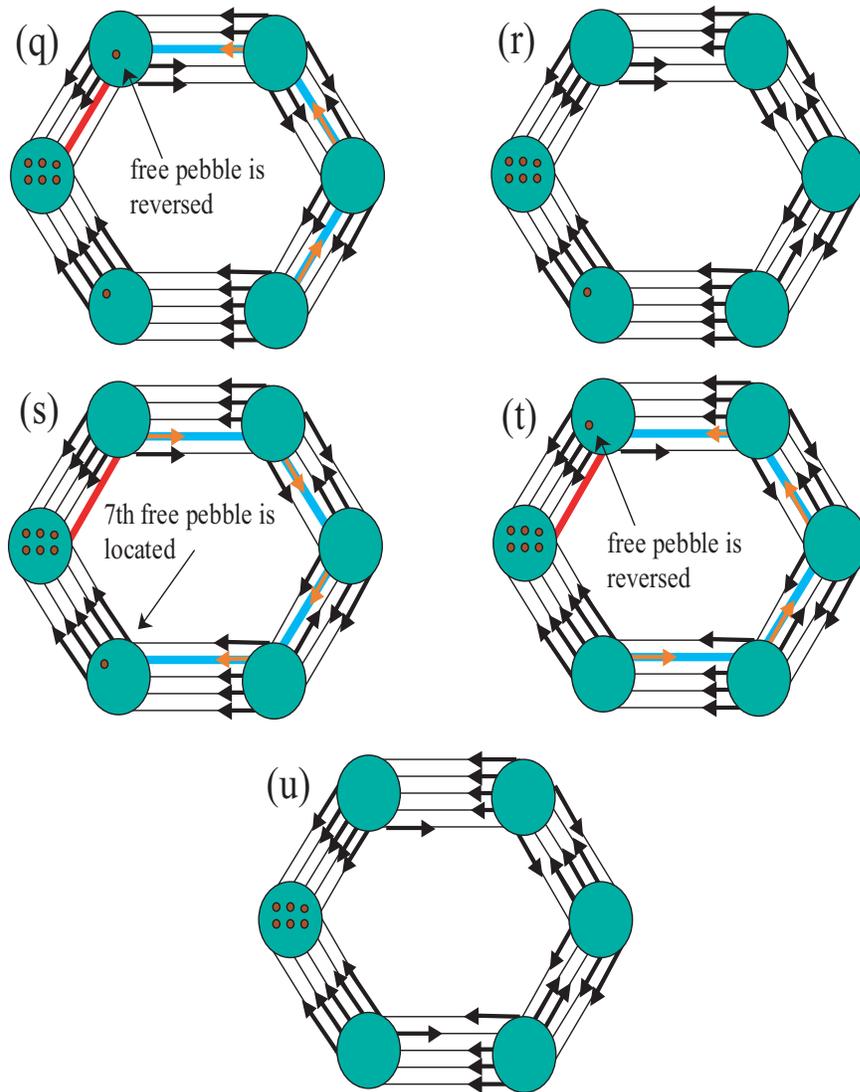


Figure 3.3: *Illustration of the $6|V| - 6$ pebble game algorithm ... continued.* We continue to test edges one by one, reversing free pebbles when needed, until all edges are tested. In this case all edges are successfully pebbled (independent). There are 6 remaining free pebbles indicating that the multigraph is rigid (i.e. has only 6 trivial DOF).

these extra free pebbles. This notion of restricting how free pebbles in certain regions can move throughout the multigraph is a very important concept, and we will look at it more deeply later on in the chapter. Having finished the pebble game, we observe that the multigraph is flexible as it has more than six free pebbles remaining. It has

ten free pebbles, indicating that it has 4 internal DOF ($= 10 - 6$). Four additional properly placed edges (independent) would be required to remove the additional free pebbles and to make the multigraph rigid.

A very useful property that can be extracted from the pebble game occurs whenever we encounter a failed search (step 2 (b) (ii) of the Algorithm). The region in the graph that the failed search traverses, which we call a *failed search region* (i.e. the subgraph spanned by the set of vertices in a failed search region), is a rigid subgraph. Moreover the failed search region is a redundantly rigid subgraph (see [172] Lemma 5.2.1). In Figure 3.5 a failed search region is shown (a multigraph of a ring of five with (blue) ⁶ vertices is redundantly rigid).

We recall from the earlier discussion that the pebble game can also decompose a multigraph into a set of maximal *rigid clusters* (i.e. maximally rigid subgraphs). This is essentially performed using failed searches. A set of vertices on V' vertices in a failed search region already contains $6|V'| - 6$ independent edges (i.e. $6|V'| - 6$ edges are covered by pebbles), and has no more free pebbles to give up. Rigid regions can have a maximum of six free pebbles on its vertices (i.e. maximum pebbles we can recover to any rigid subgraph is six) [172]. This fact should be clear, because if the rigid region had more than six free pebbles, we could add another independent edge consuming the seventh free pebble (but rigid regions have maximum number of independent edges - see also Lemma 3.2.3). The details of finding the rigid clusters also referred to as *rigid cluster decomposition*, with various computational speed ups, can be found in [79, 84, 109, 114, 115]. Rigid cluster decomposition is implemented in FIRST, which finds all the rigid clusters of a protein (i.e. in its underlying multigraph) [84, 109]. The entire multigraph in Figure 3.1 is one rigid cluster. In Figure 3.5 the

⁶A ring of five atoms occurs in a proline amino acid, for instance.

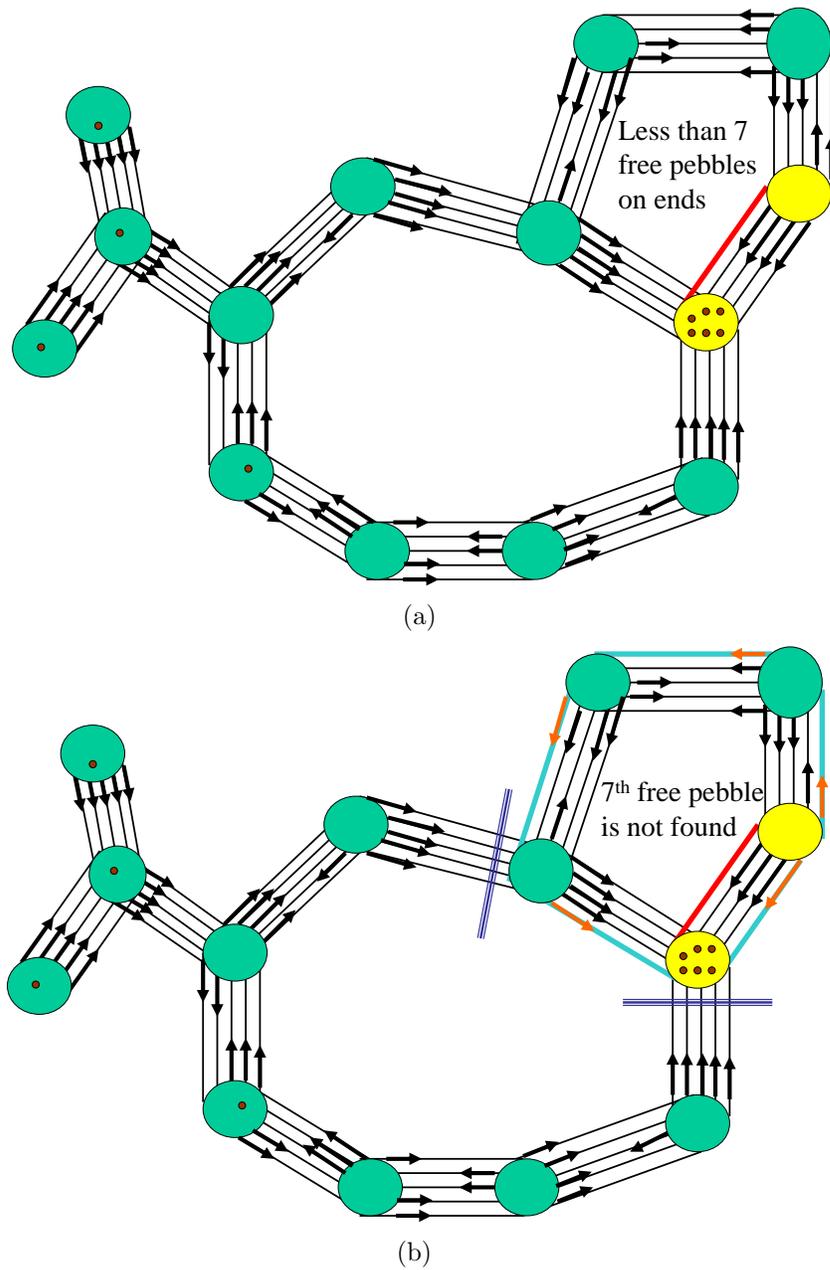


Figure 3.4: *Illustration of the $6|V| - 6$ pebble game algorithm on G with redundancy. The red edge has less than seven pebbles on its ends (a), and the seventh free pebble is not found along the directed paths from its ends (b). The rest of the multigraph has free pebbles but no path leads us these free pebbles. Continued in Figure 3.5.*

subgraph of a ring of five is one rigid cluster, and the remaining vertices (rigid bodies with 6 trivial DOF) form individual rigid clusters.

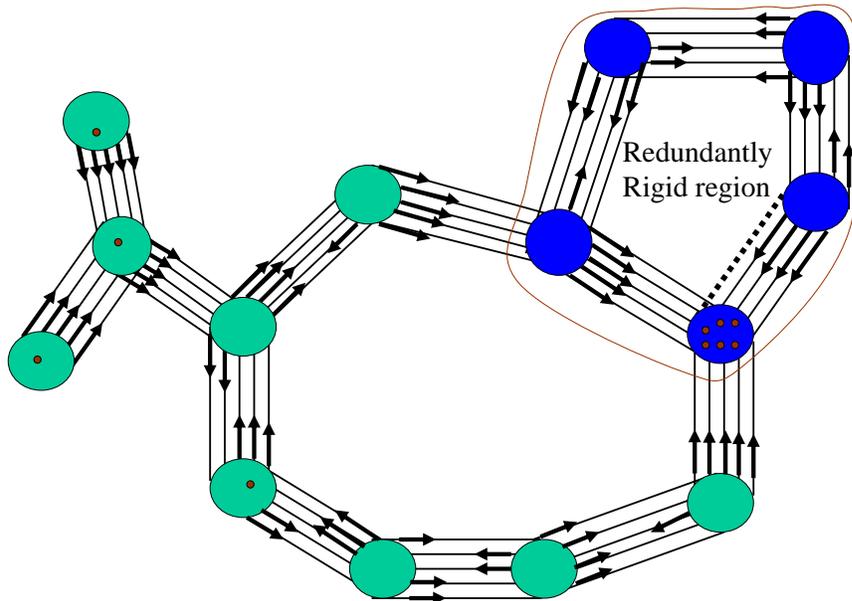


Figure 3.5: *Illustration of the $6|V| - 6$ pebble game algorithm with a redundant edge ... continued.* The failed search region locates a redundantly rigid subgraph.

From the two examples, we can already observe that the physical picture alone from the output of the pebble game algorithm provides visually rich insights into the rigidity/flexibility of the underlying multigraph.

We state a few other important observations and properties. The pebble game algorithm is a *greedy* algorithm [114, 115, 207] in the sense of computer science ⁷. In other words, we can test edges (i.e. attempt to place a pebble - test for independence)

⁷Greedy algorithms are often studied in computer science [31]. They are also well studied in the branch of combinatorics, called “matroid” theory [150], whose underlying combinatorial structures always give nice greedy algorithms [31]. Many concepts and vocabulary in matroid theory comes from linear algebra and graph theory. Basically, a matroid is a structure that captures the essence of “independence” and generalizes linear independence in vector spaces. The count that the pebble game is tracking ($6|V| - 6$) defines an independence set in a matroid (see [114, 203]). In the spirit of linear algebra, if we think about the edges of the multigraph as the rows of the matrix, the process of finding the maximal linearly independent rows is also greedy [207]. More specifically, no matter how we row-reduce the matrix (using elementary row operations), the number of non-zero rows (i.e. rank of the matrix) is always the same. Regardless of which set of linearly independent rows we choose for the basis of the row space, all bases will have the same size. The fact that all bases (a set of maximum linearly independent vectors) of the vector space have the same size is the basic model we can use in understanding greedy.

in any order ⁸ and be sure that we will always end up with the same number of pebbled (independent) edges (i.e. $|I(G)|$ is unique under different plays of the pebble game). Equivalently, since the total number of pebbled edges is unique (and as we always start with the same number of pebbles $6|V|$), the number of remaining free pebbles $6|V| - |I(G)|$ (i.e. degrees of freedom) is also unique. Other properties such as the rigid regions located and the number of redundant edges in any maximally rigid subgraph also do not depend on the order of testing edges [115].

One of the most important features of the pebble game algorithm, which is very useful in applications, is that it is extremely fast and numerically stable. The pebble game algorithm runs in time $O(|V|^2)$ and in practice the algorithm scales nearly linearly in the number of vertices [84, 114, 115]. The pebble game is an integer algorithm (exact algorithm – has no round-off errors). The algorithm is also well implemented [84, 115]. Several expositions of Lee and Streinu [114, 115] handle various details of the computational complexity along with the companion suggestions on data structures.

The various implementations of the pebble game algorithm have been applied on multigraphs with millions of vertices [84, 94]. Moreover, as mentioned earlier, due to its speed and efficiency, the pebble game algorithm is embedded within FIRST software. The algorithm can be executed on a typical protein or even viral capsids (which are typically composed of hundreds of copies of a protein) whose underlying multigraph contains millions of atoms (vertices) [27], in a matter of seconds.

Note that if the graph is not independent (i.e. has redundant edge(s)) then which set of edges get covered by a pebble (declared independent), similarly which edges get declared as redundant (not covered), is determined by the play of the pebble game. That is, testing edges in a different order may produce a different set

⁸Sometimes we will refer to testing edges of the multigraph G in a different order as a different play of the pebble game on G .

of independent edges. Of course, this difference only occurs on the order the pebble game is played on any redundantly rigid subgraphs. For instance, in Figure 3.5 in the redundantly rigid subgraph, any one of the 25 edges could have been declared redundant. As there is only one redundant edge in that multigraph, the edge that is tested last in that redundantly rigid subgraph will be the redundant edge. The pebble game sets aside redundant edges as they are not covered by a pebble, so redundant edges do not get assigned a direction ⁹. Of course, as the total number of pebbled (independent) edges $|I(G)|$ is unique under different plays of the pebble game, the total number of redundant edges $|\mathfrak{R}(G)|$ is also unique.

Redundance in a region can be helpful in applications. For instance, in the FIRST rigidity/flexibility of proteins, rigid regions that are redundantly rigid with significant number of redundant edges (i.e. three or more redundant edges) is an indication of robust rigidity [207]. For instance, breaking a hydrogen bond (which often break and reform – flicker [122]) or removing a hydrophobic constraint may not alter the rigidity of the region.

We now summarize some very useful pebble game properties and invariants that are maintained throughout the pebble game. These properties are particularly useful in constructing the proofs and they sharpen our understanding of the pebble game algorithm. They are also important in generating important extensions of the pebble game algorithm [172]. Our previous extensions of the pebble game algorithm will also lead to a number of other useful pebble game invariants which will be provided later in the chapter. We will just summarize the main results, without giving the proofs, as they were proven previously.

⁹The fact that redundant edges are not covered by pebbles is another instance of the visual appeal of the pebble game algorithm. Since redundant edges are not contributing to the rigidity of the multigraph, whether they are kept or removed the free pebble count (DOF) will remain the same.

Remark. We should point out that once the six pebbles are placed on each vertex, only two moves of the algorithm are responsible for all the changes in the multigraph. One move is the basic placement of the pebble onto an edge (directing the edge accordingly) (Step (a) Algorithm 3.2.1). The second move is the swap (or the sequence of swaps – i.e. a cascade) that redistributes the present free pebbles on the graph (Step (i) Algorithm 3.2.1). The remaining rules and procedures (eg. making sure that there are at least seven free pebbles on the ends before we cover the edge, or searching for the free pebble in the directed graph) are important self-checks that help us play the pebble game correctly, but they do not alter the *distribution* of free pebbles in the graph. By distribution we mean the location of the free pebbles in the graph. Understanding how free pebbles move throughout the different regions in the pebble game generated directed multigraph is an important concept, which will be looked at more closely in the next section, and will play an important role in applications in the subsequent chapters. ■

We say that an edge e which is incident to v is an *outgoing edge* of vertex v if that edge is pebbled and directed out of v (i.e. visually on the graph there is an arrow pointing out of v). Similarly we can extend to vertex sets, if $S \subseteq V$, the outgoing edges out of S are the set of directed edges (pebbled) from S to its complement $V - S$. Some of the following properties were already discussed above, others come from [172, 114].

Lemma 3.2.3 *Invariants of the Pebble Game Algorithm.*

Let $G = (V, E)$ be a multigraph (no loops, edge-multiplicity at most six). At every stage¹⁰ of the pebble game algorithm the following invariants are maintained:

¹⁰By “stage” we mean some “instance” of the pebble game algorithm. We are testing the edges in G , we stop the pebble game algorithm and want to discuss the current generated directed multigraph and free pebbles on the vertices.

1. For each vertex, the number of free pebbles (pebbles lying on that vertex) plus the number of outgoing edges is exactly six (i.e. all six pebbles remain associated with each vertex).
2. There are at least six free pebbles on $V(G)$.
3. For any vertex $v^* \in V$, we can recover all six (associated) pebbles back to v^* (i.e. six free pebbles appear on v^*).
4. Every subset S of $|V'|$ vertices spans at most $6|V'| - 6$ (pebbled) edges.
5. For every subset S of $|V'|$ vertices, (the number of free pebbles on vertices of S) + (the number of pebbled edges in S) + (the number of outgoing edges out of S) = $6|V'|$.

1. and 2. are obvious observations, the proof of 3. we have shown in [172] (Lemma 3.3.3) using a simple induction on the number of pebbles moves, and proofs of 4. and 5. are provided by Lee and Streinu in [114].

3.3 The Relevant Regions

“In every block of marble I see a statue as plain as though it stood before me, shaped and perfect in attitude and in action. I have only to hew away the rough walls that imprison the lovely apparition to reveal it to the other eyes as mine see it.”

– Michelangelo Buonarroti

In this section we describe several extensions of the pebble game algorithm which answer additional important queries. We will primarily focus on the *relevant regions detection algorithm*, which is an essential algorithmic component in protein

applications in chapters 4 and 5. We will provide a short description and state some important properties, where more detailed discussion can be found in [172].

3.3.1 Quantifying relative flexibility of the core

Before we can explain the concept of relevant regions, we need to be able to quantify the (relative) flexibility of a subgraph in G .

In many situations and applications (see chapters 4 and 5), we are not interested in the overall flexibility of a multigraph, but rather of a certain subgraph (region) or several (disjoint) subgraphs within the larger multigraph. In order to quantify the flexibility of the subgraph(s) we need to carefully track the possible distributions (i.e. locations) of the remaining free pebbles in the pebble game generated directed multigraph.

Remark. To erase any possible future confusion with notation, it should be recalled that G is an undirected multigraph. However, when we refer to the output of the pebble game on G (similarly with G_c), the edges that are pebbled are certainly directed and this gives a (pebble game generated) directed multigraph. Similarly, when we are referring to the directed multigraph, directed edges, outgoing edges, etc., it should be clear that the pebble game has been played on the starting undirected multigraph G . Thus, there should be no uncertainty when we are talking about a directed or undirected multigraph. ■

Given a multigraph G (edge multiplicity at most six), we define the *core* G_c of G as a nonempty (induced) subgraph of G ($G_c \subseteq G$). We are interested in quantifying the flexibility of the core. More specifically, we want to know the total number of *relative DOF* of the core. If we ignore the 6 trivial DOF, the (internal) relative DOF of the core is the number of independent edges that must be added to G_c (core) to

make it a rigid region ¹¹. We chose the term ‘relative’ as the rest of the multigraph ($G \setminus G_c$) can constrain (restrict) and reduce the DOF (motions) of the core. That is, if $G \setminus G_c$ is removed, there might be an increase in DOF of G_c . This concept is central to the ‘relevant regions’, which will be discussed shortly. For brevity, we may sometimes drop the prefix ‘relative’, where it is understood we are always discussing rigidity and flexibility of the core in the presence of the rest of the multigraph.

Quantifying the relative DOF of the core is easily obtainable from the output of the pebble game algorithm. The basic procedure is to play the pebble game on the entire multigraph G and once the pebble game is finished, we draw as many free pebbles as we can to the vertices in the core (i.e. to $V(G_c)$). The maximum number of free pebbles that we are able to reverse is the total relative DOF of the core. We now present the pseudocode of this procedure [172].

In the output of the pebble game, we say that an edge is *outgoing* from G_c , if it directed out of G_c . That is, in the outgoing edge out of G_c , the initial vertex is in $V(G_c)$ and terminal vertex is in $V(G) \setminus (V(G_c))$.

Algorithm 3.3.1 – Draw Maximum pebbles algorithm

Input: multigraph $G = (V, E)$, and core $G_c = (V_c, E_c) \subseteq G$.

Output: the maximum number of free pebbles on G_c .

1. Play the pebble game algorithm on G (Algorithm 3.2.1) and freeze ¹² the free pebbles on the vertices in the core (V_c).
2. If there are any outgoing edges out of G_c (i.e. arrows directed out of G_c), then proceed to step 3., otherwise go to step 6.

¹¹Note that the general definition of the the core is any induced subgraph in G , however, in applications the interesting cases will usually be when the core is defined as two disjoint induced subgraphs. For instance, in hinge motions predictions in proteins (Chapter 4), the core will be formed of two separate large rigid clusters.

¹²The free pebbles that are frozen on the vertices in the core are not allowed to be swapped (cascaded) away from their vertices.

3. Consider any vertex $v_c \in V_c$ which is incident with an outgoing edge from G_c . Attempt to find a free pebble by following a directed path (covered edges), never searching over any vertices in V_c (as pebbles in V_c are frozen).
 - (a) If the free pebble is found on some vertex (not considering the frozen pebbles on V_c), then draw the free pebble back as a cascade to v_c (see step 2. (b) (i) of Algorithm 3.2.1).
 - (b) Else, the free pebble is not found, and we have a capped failed search ¹³, proceed to step 5.
4. If there are more outgoing edges out of v_c in step 3, then return to step 3. Otherwise, proceed to step 5.
5. If there are more untested vertices with outgoing edges out of G_c , return to step 3. Otherwise proceed to step 6.
6. Stop, and count the number of free pebbles on V_c .

In Figures 3.6 – 3.8, we have illustrated the ‘Draw maximum pebbles’ Algorithm 3.3.1. Once the pebble game algorithm is completed on G , we look for any vertices in G_c that have outgoing edges (yellow vertices with red outgoing arrows – Figure 3.6 (b)). The entire multigraph has 11 free pebbles, and only 5 are on G_c , so we should clearly be able to recover more free pebbles to G_c (Lemma 3.2.3 3. and 5.).

Starting from a yellow vertex, we search for a free pebble and recover (draw) it back to G_c as a cascade (Figure 3.7 (a) and (b) and Figure 3.8 (a) – Step 3 (a) of Algorithm 3.3.1). We continue to attempt to draw free pebbles back to G_c one at a time. Note that every time a free pebble is recovered to the core, we have reversed the

¹³It is important to point out that the capped failed search is not the same as the regular failed search (step 2. (b) (ii) of Algorithm 3.2.1) that we encounter when we are not able to recover seven free pebbles to the ends of the edge we are testing.

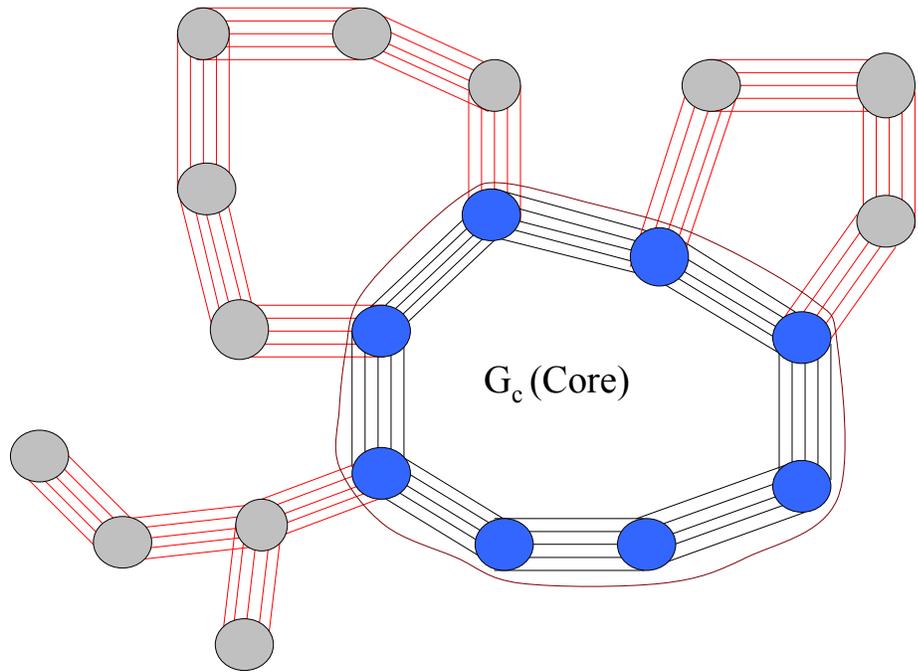
orientation of the directed path (indicated in turquoise). In Figure 3.8 (b) the search out of the core is not successful as free pebble is not found (step 3. (b) of Algorithm 3.3.1), and we have a *capped* failed search. The reason we call it a capped failed search is that we are not allowing the search for a free pebble to go over the vertices in V_c , otherwise we would be only redistributing the free pebbles that are already in the core.

In this example we are able to recover 7 free pebbles to the core G_c , indicating that the core has a total of 7 relative DOF, equivalently 1 internal DOF. So, we need to add only one edge to the core, to make it a rigid region. On the other hand, the entire multigraph has 11 additional free pebbles, so we would need to add 5 (independent) edges to make the whole multigraph rigid.

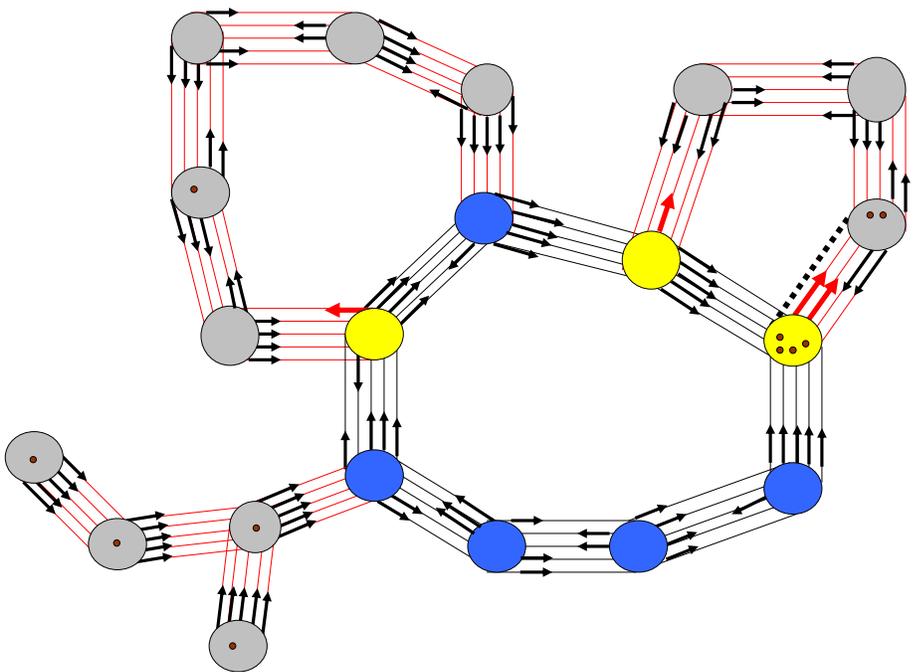
Once we have recovered the maximum number of free pebbles to the core (i.e. determined the relative DOF) the natural desired property is that we will always get a unique number of free pebbles regardless how we played the pebble game algorithm 3.2.1 on G (i.e. the order the edges were tested). That is, we want to know if the greedy property of the pebble game extends to the maximum number of free pebbles on any subgraph. This is captured in the following theorem (Theorem 5.1.1 [172]):

Theorem 3.3.2 *Let $G = (V, E)$ be a multigraph, and $G_c \subseteq G$. The maximum number of free pebbles we can recover to G_c is the same for every complete play of the pebble game algorithm on G . (i.e. The output of the Algorithm 3.3.1 is invariant under different plays of the pebble game.)*

The full proof of this theorem is quite lengthy and requires several lemmas and other pebble game invariants. All the details can be found in [172].

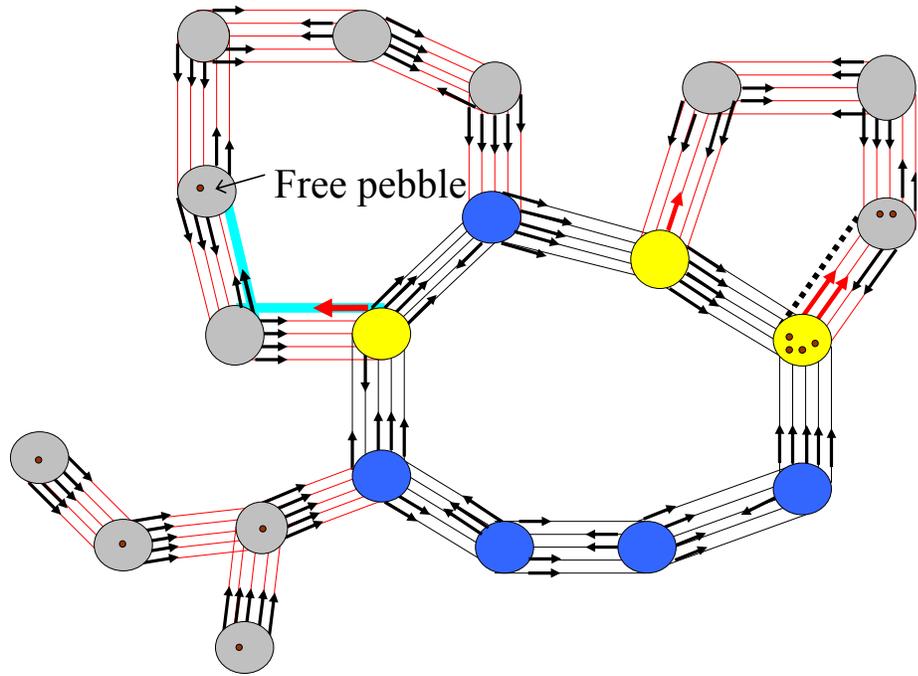


(a)

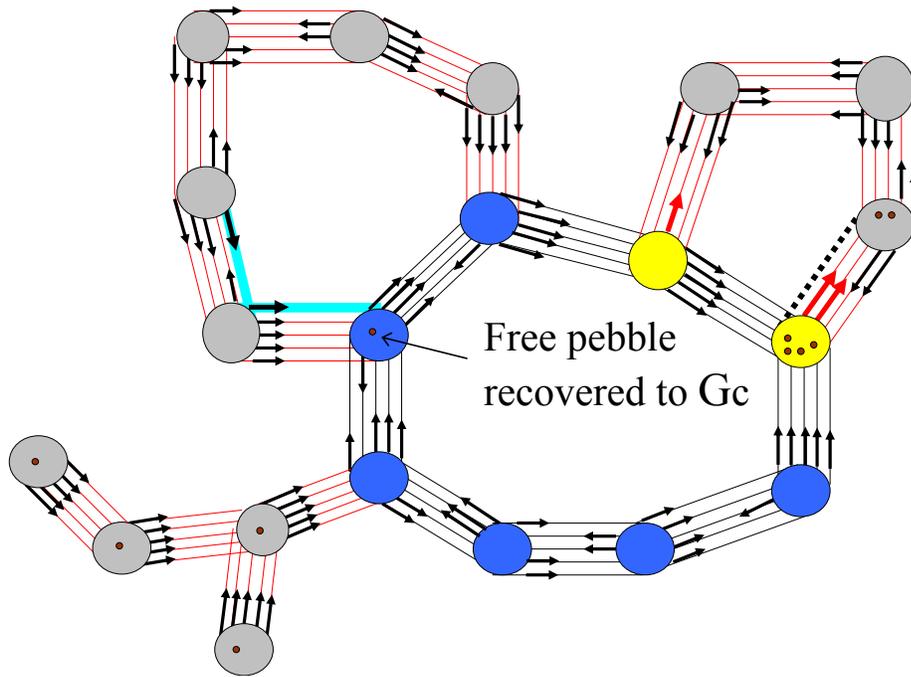


(b)

Figure 3.6: *Illustration – drawing back free pebbles to the core.* Once the pebble game is completed on the entire multigraph, we find all the outgoing edges (b) out of the core (shown as red arrows). Continued in the next figure ...

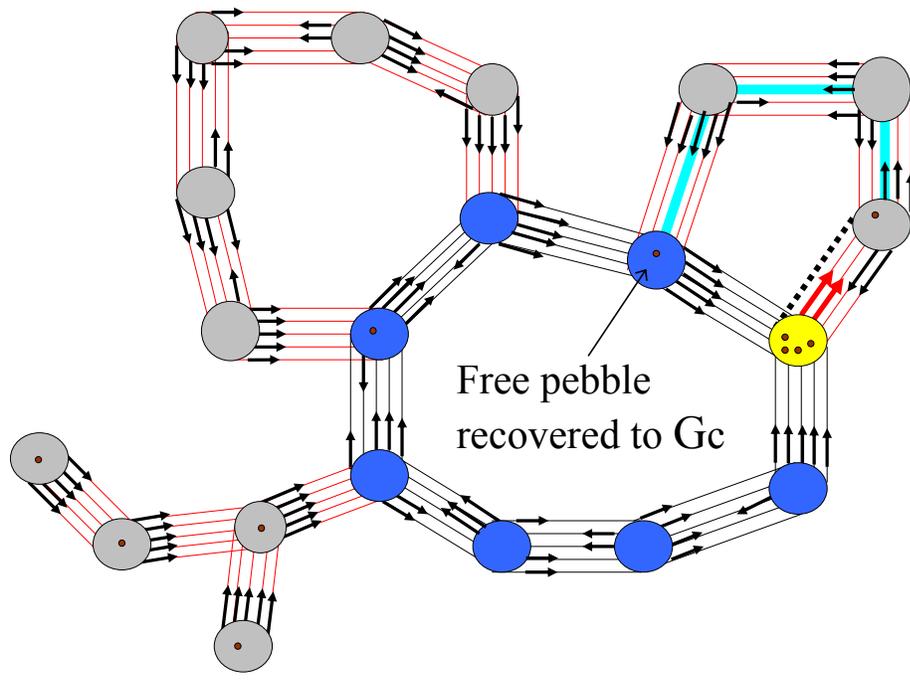


(a)

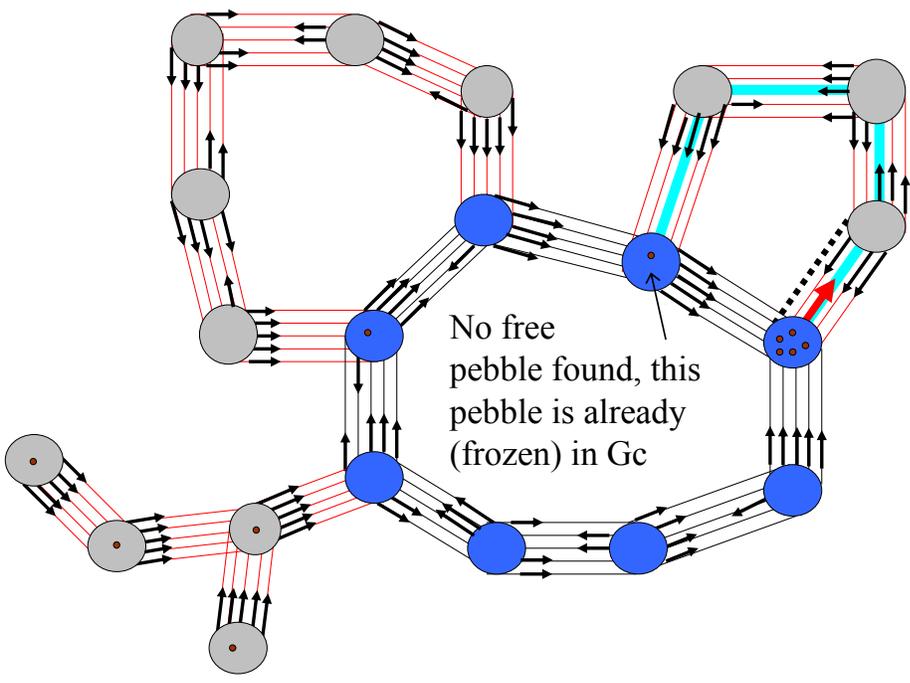


(b)

Figure 3.7: Continued from the previous figure... We search out of the core for a free pebble (a) and reverse a free pebble as a cascade (b). Continued in the next figure ...



(a)



(b)

Figure 3.8: Continued from the previous figure... We continue to reverse free pebbles one by one, back to the core (a). In (b) we have recovered the maximum 7 free pebbles to the core. The core has a total of 7 relative DOF, or 1 internal DOF.

3.3.2 Relevant region detection

As we will shortly see, drawing back of maximum free pebbles to the core (Algorithm 3.3.1) is an important first step in the detections of relevant regions, which we turn to next.

We have already given a hint that regions outside of the core can constrain and impact the possible motions (i.e. reduce DOF) of the core. To motivate the idea of relevant regions, it should be intuitively clear that if we are interested in the relative DOF count of the core G_c , there could exist a set of vertices and edges in $G \setminus G_c$ that we can remove (prune) from G and still be left with the same number of relative DOF of G_c . In contrast, removal of some other vertices and edges in $G \setminus G_c$ can increase the relative DOF count of G_c . Regions in $G \setminus G_c$ whose removal increase the relative DOF count of G_c are called *relevant*, otherwise they are *irrelevant*. The relevant regions are responsible for reducing the DOF count (possible motions) of the core, and irrelevant regions have no impact on the motions of the core.

The relevant regions can be nicely captured via an extension of the pebble game algorithm, and will be explicitly defined from the output of the Relevant region detection algorithm 3.3.3.

As pebbles are markers for degrees of freedom, the pebble game algorithm can be useful to attain a deeper understanding of the concept of relevant regions. Once we have recovered the maximum free pebbles to the core G_c , removal of a relevant region will increase the maximum number of free pebbles on the core. Relevant regions permanently remove some free pebble(s) that are associated with the vertices of the core. In other words, the presence of a relevant region with respect to the core, means that even after the maximum free pebbles are recovered to the core, some free pebble(s) associated with the vertices in the core are used to cover edge(s) outside of the core. Since every vertex retains all of its 6 associated free pebbles, any free

pebble that is permanently removed from the core will be used to cover an edge that is outgoing out of the core. So, an important indication of the existence of a relevant region is an outgoing edge(s) out of the core (once maximum free pebbles are recovered to the core). Each outgoing edge corresponds to a single removed DOF (free pebble) from the core. The outgoing edges out of the core are central in detecting relevant regions. More specifically, the outgoing edge(s) out of the core lead to a capped failed search, which identifies a relevant region. The capped failed search region is a relevant region since removal of any covered edge(s) (or vertex) in the capped failed search region, would increase the number of free pebble(s) at the core. The free pebble can be reversed back to the core using a directed path in the capped failed search region.

In contrast to the relevant regions, removal of irrelevant regions leaves the same maximum number of free pebbles on the core. Once the maximum free pebbles are recovered to the core, an indication that everything outside of the core is irrelevant is the lack of any outgoing edges out of the core.

Our goal is to decompose the multigraph into two regions, those that are relevant with respect to some predefined core G_c , and other regions that are irrelevant. Since we will partition G into two regions (relevant and irrelevant), we will naturally want to include the core G_c to the relevant region. Obviously, removal of pebbled edges in the core can increase the DOF (free pebbles) in the core, so the core is in the most trivial sense relevant to itself. The image is that we start with the core and add to it a relevant region, obtaining an *enlarged relevant region*, which we will denote as G_R (see also Figure 3.9). The remaining part of the multigraph (i.e. $G \setminus G_R$) will be declared irrelevant. However, because of the inherent triviality of the core as a relevant region to itself, generally speaking when we are referring to the relevant region, we are primarily referring to any relevant region outside of the core (i.e. $G_R \setminus$

G_c). If $G_c = G_R$, (i.e. no pebbles (DOF) are removed outside of the core) we will say that the core has no relevant region (i.e. everything outside the core is irrelevant).

We now present an algorithm which captures the relevant region of the core.

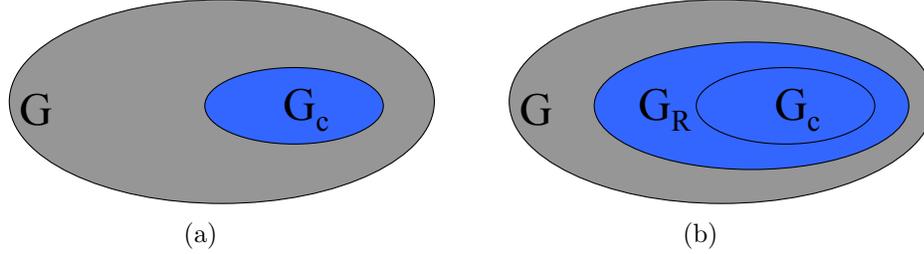


Figure 3.9: *Relevant region decomposition*. Given a multigraph G , and some core G_c . We identify a relevant region outside of the core, and expand the core to an enlarged relevant region G_R . The remaining part of G is irrelevant and coloured gray.

We will make use of this definition on the pebble game generated directed multigraph. At any instance (stage) of the pebble game algorithm, we define the *Reachability region* $Reach(v)$ ¹⁴ of a vertex v as all the vertices w such that there exists a directed path from v to w (i.e. w are all the vertices that can be reached from v).

Algorithm 3.3.3 – Relevant Regions Detections Algorithm

Input a multigraph $G = (V, E)$, and core $G_c = (V_c, E_c) \subseteq G$.

Initialize R as an empty set of vertices.

Output a set of relevant vertices R , outside the core G_c , and the enlarged relevant region G_R with respect to the core G_c .

1. Play the pebble game on G (Algorithm 3.2.1), and draw the maximum number of free pebbles to G_c (Algorithm 3.3.1).

¹⁴We exclude paths of length zero.

2. If there are any outgoing edges from G_c (arrows directed out of G_c), then go to Step 3, otherwise ¹⁵ proceed to Step 4.
3. For every vertex $v_c \in V_c$ which is incident with an outgoing edge from G_c (i.e. there is an arrow pointing out from G_c at v_c) do steps (a) and (b).
 - (a) Find $\text{Reach}(v_c)$, never searching over any vertices in the set V_c .
 - (b) Add $\text{Reach}(v_c)$ to set R , excluding any vertices that are already in R .
4. Output set R (a set of relevant vertices outside of the core G_c).
5. Define $V_R = R \cup V_c$. Complete the algorithm by outputting the enlarged relevant region $G_R \subseteq G$ (which includes G_c), where $G_R = (V_R, (V_R \times V_R) \cap E) = (V_R, E_R)$.

The most important component of the Algorithm 3.3.3 is to identify a set of relevant vertices R (steps 2 – 4) for the core G_c . Starting with the predefined core G_c , taking the set R , we then grow (enlarge) the core G_c to an enlarged relevant region G_R (step 5). Once we have obtained the enlarged relevant region G_R , we have decomposed the multigraph G into two regions, G_R (relevant + core), and $G \setminus G_R$ the irrelevant regions. See Figure 3.9 for the schematic representation of the relationship of the multigraph G , the core G_c and the enlarged relevant region G_R .

Note in step 3.(a) of the algorithm, as we look for the reachability region, we do not need to search for relevant vertices R from every outgoing edge of v_c . In other words, if there is more than one outgoing edge from the core at the same vertex v_c , it is enough to only consider a ‘single’ outgoing edge from v_c , because other outgoing edges from this vertex will locate the same reachability region. It is also important to realize that the reachability region in step 3.(a) leads to a capped failed search region

¹⁵If there are no outgoing edges out of G_c , every single pebble associated with the set of vertices belong to G_c is either being used to cover some edge in G_c or it is a free pebble in G_c .

that we had discussed in Algorithm 3.3.1, when we had more outgoing edges out of the core and the search did not find any more free pebbles.

It should be clear that a single outgoing edge from G_c (after we have recovered the maximum pebbles to G_c) guarantees that some region outside of the core G_c is relevant (i.e. R contains at least one vertex). On the other hand, when there is no outgoing edges from G_c , then everything outside the core is irrelevant (i.e. R is an empty set – it contains no vertices), and the algorithm will output that $G_R = G_c$. When there is no relevant region outside of the core, the relative DOF of G_c (i.e. maximum free pebbles) would remain the same if we removed all the vertices and edges in $G \setminus G_c$.

In the spirit of Michelangelo’s quote, as a poetic description of the relevant region algorithm, we would like to throw away all the irrelevant regions of the core, and be left only with core and its relevant regions.

Remark. The initial motivation for the development of the relevant region detection algorithm was stimulated by the outputs generated by program FRODA [48, 199] (an extension of FIRST, which simulates the initial motions of the protein) on protein immunoglobulin. We observed that too much time and effort was devoted in simulating pieces that were not important to the motions of the regions (core) that we were interested in. Perhaps it is intuitively clear that one would need to remove loose dangling ends, or even long flexible loops attached to the core, as they are not contributing to the rigidity of the core (i.e. they are irrelevant). However, by using the Algorithm 3.3.3, as we have a complete answer, we can also locate numerous not so obvious irrelevant regions. In this selective course graining, we capture the *relevant regions* as they constrain and impact the possible motions and flexibility (DOF) of the *core*. On the other hand, we could remove and neglect the *irrelevant regions* as flexibility of the core would remain the same. By doing this we can potentially

remove hundreds of degrees of freedom (associated with irrelevant regions), and deal with significantly lower number of (i.e. relevant) degree of freedom. We would not be spending time and computational resources simulating the motions of irrelevant pieces. ■

We will outline several important properties of the output of the Relevant region detection algorithm, but first we need to illustrate these ideas with an example. As we have already seen, visual display of any algorithms involving the pebble game, such as the relevant region detection is very important as it significantly sharpens our understanding of the concepts and the corresponding algorithms.

As a simple illustration of the Relevant Regions Detection Algorithm 3.3.3, let us revisit the example in Figure 3.6. Relevant region detection of the core G_c is shown in Figures 3.10 and 3.11. In Figure 3.10 (a) the pebble game was already played and the maximum number of free pebbles are recovered to the core (step 1. of algorithm). First of all, we observe that the core is a ring of size 8, which as an isolated multigraph should have 8 free pebbles, equivalently 2 internal DOF (recall also Figure 2.13). In this example with the presence of the rest of the multigraph G , we were able to recover only 7 free pebbles. This indicates that one free pebble (DOF) is removed from the core, and there exists a relevant region outside the core that is constraining the motions of the core.

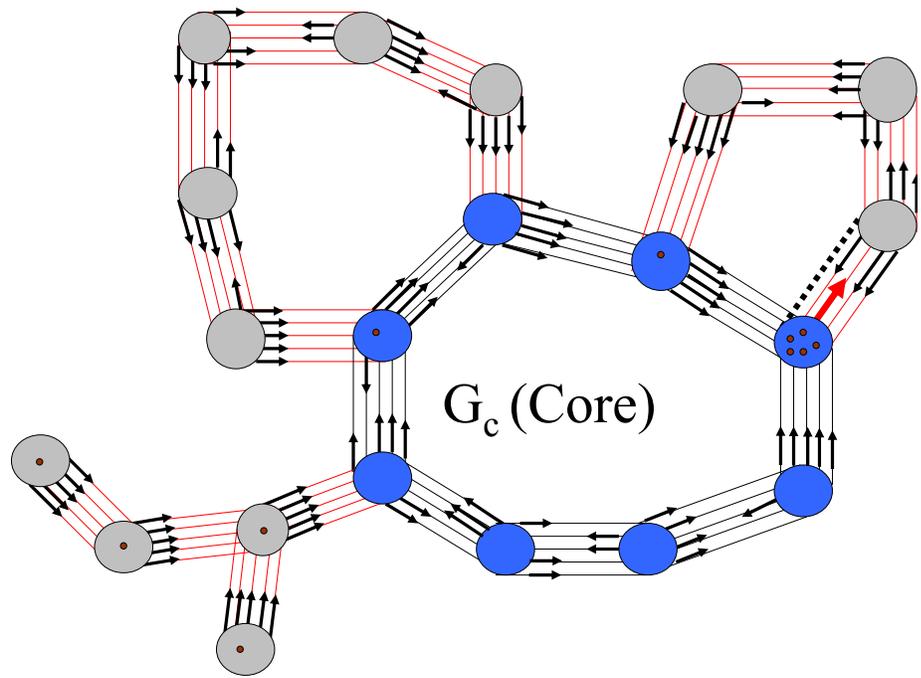
We note that there is a single outgoing edge out of the core (depicted with a red outgoing arrow – corresponding to the permanently removed free pebble from the core). The three red vertices define the set R (Figure 3.10 (b)) which is the reachability region of the vertex incident with the red outgoing arrow (step 3. (a) of algorithm). The corresponding capped failed search region gives us the relevant region of the core. Note how the rest of the multigraph is irrelevant – there exists no capped failed search out of the core (i.e. no outgoing edges out of the core). In

Figure 3.11, we have combined the core and the relevant region into the enlarged relevant region G_R , with the irrelevant region coloured in gray. Removal of the entire irrelevant region would still leave the same number of free pebbles at the core (7 in this example). In terms of proteins, we can think of the two parts of the irrelevant region in this example as a flexible dangling end (i.e. side chain that is just abstractly connected to the core and not participating in hydrogen bonding) and a flexible loop. Removal of both has no impact on the motions (DOF) of the core. On the other hand, the relevant region is essential to the core, as its removal would increase the number of free pebbles at the core from 7 to 8.

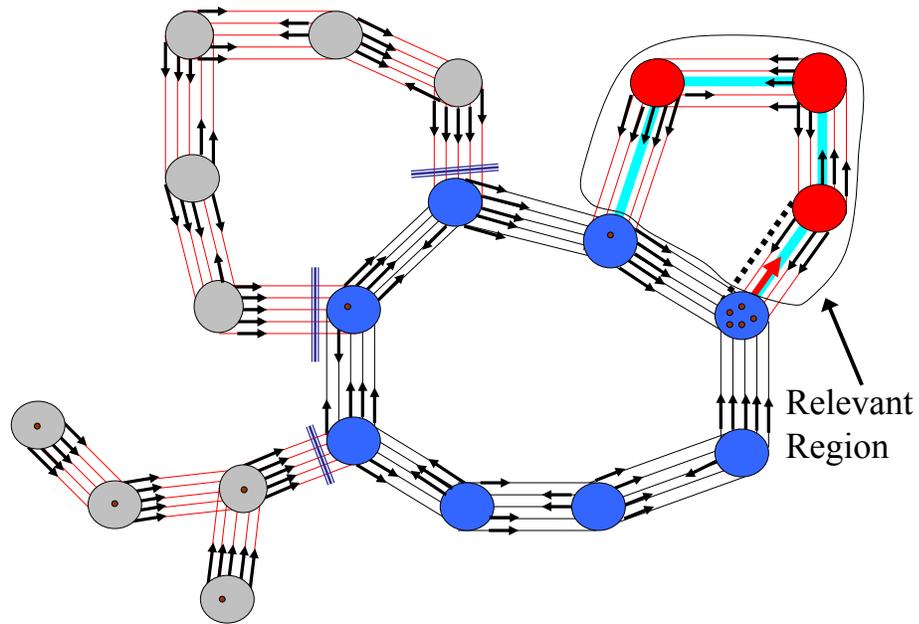
More examples on relevant region detection can be found in [172], and also examples in a special setting designed for hinge predictions and in allostery are provided in chapter 4 and 5.

Ambiguity

In some multigraphs, ambiguity could occur in the declared relevant region, depending on the play of the pebble game (i.e. which order the edges in G are tested) on the multigraph G . That is to say if the edges on certain subgraphs (see below for more details) are tested in a different order then two plays of the pebble game may have slightly different declared relevant regions (i.e. G_R in the output of Algorithm 3.3.3 is not unique). The ambiguity is not very significant and is a natural property of the multigraph and its rigidity, and not the algorithm that we have developed (see below for more discussion). We are mostly exposing the ambiguity for completeness and richer understanding of the algorithms and the rigidity of the underlying multigraph. Nevertheless, we will suggest some techniques that can capture all the possible relevant regions under all different plays of the pebble game. See further below in this



(a)



(b)

Figure 3.10: *Relevant region detection*. Continued in Figure 3.11. Once maximum free pebbles are recovered to the core (a), an outgoing edge out of the core leads to a capped failed search, which locates a relevant region (b).

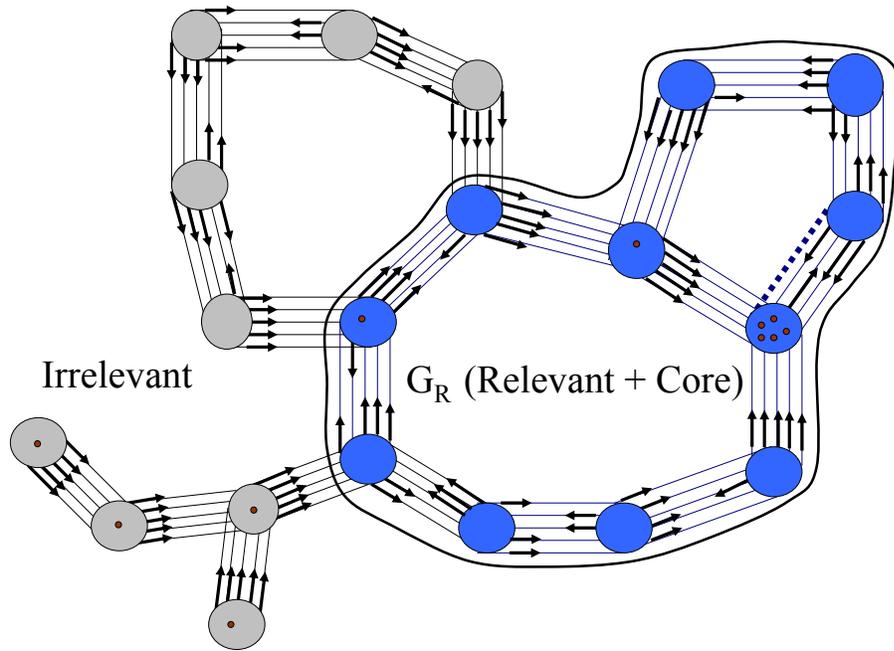


Figure 3.11: *Relevant region detection.* ... Continued from Figure 3.10. Core and the relevant region are combined into the enlarged relevant region G_R (blue). The rest of the multigraph (gray) is irrelevant.

chapter on the more important properties of the declared relevant regions that are invariant under all plays of the pebble game.

As an example of ambiguity in the relevant regions, consider the multigraph in Figure 3.12 with 5 vertices. The core is defined as the two blue vertices (a), which were made bigger for the visual clarity, and the three vertices (coloured in gray) outside the core are labelled as 1, 2 and 3. The whole multigraph G is redundantly rigid; it is heavily overconstrained (redundant) with 6 redundant edges ($6(5) - 6 = 24$ independent (pebbled) edges are needed, but there is a total of 30 edges). The maximum number of free pebbles are drawn back to the core (6 free pebbles – verifying that G is rigid), and the outgoing edges are depicted as red arrows. The capped failed search region, which finds the relevant region, is shown in (b), with the enlarged relevant region G_R (c).

If the irrelevant connection between the two core vertices (i.e. vertex 3 and its 10 incident edges) is removed, the rest of the multigraph would still remain rigid and the two core vertices would still have maximum 6 free pebbles. This indicates that the output of the relevant region detection algorithm is indeed correct. Since the three vertices and its edges are not distinguishable in the larger multigraph, it should be clear that depending on the initial play of the pebble game, any two vertices from 1, 2 and 3 (or all three) and its incident edges could come up as relevant. Regardless of which two connections are declared as relevant, removal of any one of the three vertices and their connections would still leave rest of the multigraph as rigid. That is, the number of free pebbles (DOF) at the core would remain the same (see below for generalization of this fact). In this sense, ambiguity is not a significant issue. It should be clear from this example that the ambiguity is inherent in the rigidity of the multigraph and not in the techniques or methods we are using to capture a relevant region.

Ambiguity can only occur when edges are tested in a different order on redundantly rigid subgraphs. The order the edges are tested on the redundantly rigid subgraphs, will determine which collection of edges are declared as independent (covered by a pebble) and which edges are redundant. Depending on which sets of edges are covered by a pebble on a redundantly rigid subgraphs, may lead to a different capped failed search region out of the core. More precisely, the reachability region (steps 3 and 4 of Algorithm 3.3.3 – set R) may not be unique for all plays of the pebble game.

We should emphasize that the ambiguity is not an artifact of the pebble game algorithm or the relevant region detection algorithm. The pebble game algorithm is purely a tool we are adapting to detect regions in the multigraph that are contributing to the rigidity of the core (i.e. relevant regions). Ambiguity is purely a property of the

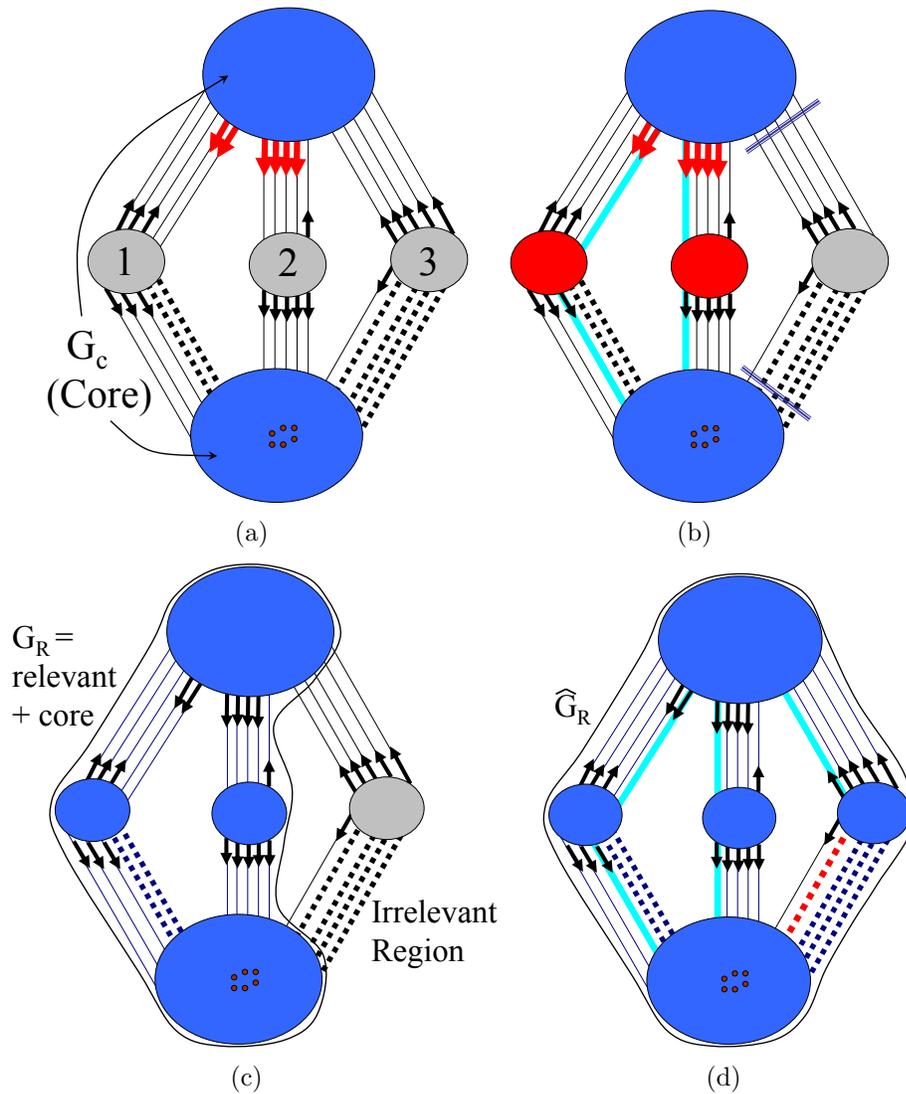


Figure 3.12: The maximum number of free pebbles are drawn back to the core (a), and the outgoing edges are depicted as red arrows. The capped failed search region, which finds the relevant region, is shown in (b), with the enlarged relevant region G_R (c). Depending on the initial play of the pebble game, any two connections (or all three) could come up as relevant. This property is inherent in the rigidity of the multigraph and not in the pebble game algorithm or the relevant regions detections algorithm. In (d) we test a redundant edge (red) in $G \setminus G_R$ for a redundantly rigid subgraph, whose failed search region includes at least one edge in G_R , so we obtain a super enlarged relevant region \widehat{G}_R . Growing the relevant region G_R to \widehat{G}_R , will remove any ambiguity as it captures every possible G_R .

underlying rigidity of the multigraph G . Regardless of the tool or method that is used to detect the relevant regions, the ambiguity would still occur. In fact, given that the ambiguity in the relevant region detection algorithm occurs, further reinforces that our relevant region detection method is a powerful and efficient technique in capturing the relevant regions.

Even though ambiguity is not critical (see Propositions 3.3.5 and 3.3.6 on more important invariant properties of the relevant region detection algorithm), if one desires, one possible way to remove the ambiguity, is to capture all possible relevant regions (i.e. under all different plays of the pebble game) and combine them into a *super enlarged relevant region* \widehat{G}_R . Since ambiguity occurs on redundantly rigid subgraphs, we grow all edges in the declared enlarged relevant region G_R to their maximum redundantly rigid subgraphs, obtaining a super enlarged relevant region \widehat{G}_R . Clearly by including more vertices and edges that are part of the same redundantly rigid region as edges in G_R , we will never introduce additional free pebbles, so we will still be left with the same free pebble count (DOF) as G_R . We conjecture that the super enlarged relevant region \widehat{G}_R contains all the possible relevant regions G_R under all plays of the pebble game (see Figure 3.12 (d)). We state this more precisely as a conjecture:

Conjecture 3.3.4 *Let $G = (V, E)$ be a multigraph, and core $G_c \subseteq G$. Assume we ran the Relevant region detection algorithm 3.3.3 and obtained the enlarged relevant region G_R with respect to G_c . For every (redundant) edge e in $G \setminus G_R$ ¹⁶, find the associated redundantly rigid subgraph, call it G_e ¹⁷. If G_e contains at least one edge in G_R , then add all vertices and edges in $G_e \setminus G_R$ to G_R ¹⁸. Continue this growing*

¹⁶Edge e has to have at least one endvertex outside of G_R .

¹⁷Recall that when we are not able to find seven free pebbles onto the ends of e (i.e. the edge is redundant), the corresponding failed search region (i.e. reachability region of the ends of e) gives us the redundantly rigid subgraph (see also [115]).

¹⁸We never include a vertex or an edge if it is already part of the current grown region.

process until all edges e have been checked ¹⁹, giving us the ‘super enlarged relevant region’ \widehat{G}_R ($\widehat{G}_R \subseteq G$) (see Figure 3.12 (d)). We have the following property:

Any two complete plays of the pebble game on G , give a unique \widehat{G}_R ²⁰.

3.3.3 Properties of the Relevant regions

As mentioned earlier the ambiguity is not a significant issue, and we have only exposed it for wider understanding. We now summarize critical properties of the output of the Relevant region detection algorithm 3.3.3 that are unchanged under different plays of the pebble game.

Proposition 3.3.5 (Proposition 5.3.1 [172]) *Let $G = (V, E)$ be a multigraph (no loops, edge-multiplicity at most six). Let $G_c = (V_c, E_c) \subseteq G$ be the core. Assume the pebble game is played on G and the maximum number of free pebbles is recovered to G_c as per Algorithm 3.3.1. Let $G_R \subseteq G$ be the enlarged relevant region (with respect to the core), which is found using the Algorithm 3.3.3. The following two properties are true for each play of the pebble game on G :*

- (i) *There are no free pebbles in $V_R \setminus V_c$.*
- (ii) *There are no outgoing edges from G_R .*

The proof of this proposition is trivial (see [172] for details). The next proposition, follows immediately from Theorem 3.3.2 and Proposition 3.3.5,

Proposition 3.3.6 (Proposition 5.3.1 [172]) *Given two complete plays A and B on multigraph G . Assume that the maximum number of free pebbles is recovered to the*

¹⁹Note if an edge was already included to the grown subgraph, it does not have to be tested. Similarly, if a redundantly rigid subgraph G_e does not share any edges with G_R , all other redundant edges in this subgraph do not have to be tested.

²⁰Play A gives enlarged relevant region $G_{R(A)}$ with super enlarged relevant region $\widehat{G_{R(A)}}$, play B gives enlarged relevant region $G_{R(B)}$ with super enlarged relevant region $\widehat{G_{R(B)}}$, then $\widehat{G_{R(A)}} = \widehat{G_{R(B)}}$. That is two people running the pebble game, may not get identical G_R , but \widehat{G}_R is always the same.

core G_c in both A and B. Let $G_{R(A)}$ and $G_{R(B)}$ be the enlarged relevant region found in play A and B, respectively, using the procedure described in Algorithm 3.3.3. The maximum number of free pebbles on $G_{R(A)}$ and $G_{R(B)}$ is the same.

This proposition indicates that ambiguity in the relevant regions is not important, as it guarantees that under all plays of the pebble game, the maximum number of free pebbles on the enlarged relevant region G_R is always the same. So, if the irrelevant region (i.e. $G \setminus G_R$) is removed/pruned (gray region in Figure 3.9), we are always left with the same free pebble count (i.e. the motions (DOF) of one enlarged relevant region are always identical to the motions of another enlarged relevant region). If we remove the irrelevant regions and try to simulate the motions of G_R (relevant + core), we are always dealing with the exact same number of degrees of freedom.

For completeness, we also summarize this result giving us an important invariant of the distribution of free pebbles in the output of the pebble game:

Theorem 3.3.7 (*Theorem 5.3.3 [172]*) *Let G be a multigraph. Consider any two (arbitrary) complete plays of the pebble game, play A and B on G . If the location (distribution) of free pebbles on the output of A and B is not the same, we can attain the same location of free pebbles by simply drawing (as a cascade) free pebbles on the output of A until the location of free pebbles is identical to the output of B.*

In this chapter we have provided a detailed explanation of the pebble game algorithm, outlined several important properties and illustrated the pebble game algorithm (see also Chapter 7). We have also described an important extension of the pebble game, the relevant region detection algorithm.

We now turn to important applications in proteins, where we extend and refine the relevant regions detections and apply it to the hinge prediction in proteins, and to protein allostery in Chapter 5.

Chapter 4

Hinge motion predictions

4.1 Overview

Understanding protein motions offers a straight-line connection between its structure and function. Normally protein motions are very difficult to study and predict as was discussed in the first Chapter. A considerable number of proteins function by domain *hinge bending motions*, or in short *hinge motions*, which involves a rigid region or domain of the protein moving relative to another rigid region about a hinge connecting the two rigid regions [61, 62, 81]. Hinge motions, and domain motions in general, play an integral role in many biological processes, including binding to other proteins and ligands, catalysis, drug docking, folding rates, etc. [81, 134]. Hinge motions typically involve large-scale conformational changes, and even for proteins of moderate size, hinge bending motions usually take place on time scales which are largely inaccessible to Molecular Dynamics simulations [40]. Given a single snapshot structure of the protein (i.e. pdb file), a first natural step in understanding the hinge motions is to be able to predict the hinge location.

In [172] we had briefly indicated that one of the applications of the Relevant Region detection algorithm is detection of hinges. In this chapter we extend and

refine the Relevant Regions detection algorithm and develop a novel Hinge-prediction algorithm as an add-on in FIRST that can accurately detect hinge locations in hinge-bending proteins. The hinge-prediction algorithm and applications to numerous hinge bending proteins is new original work.

In order to properly introduce the methodology of this algorithm and a more in-depth discussion of these concepts, we will first look at some key simple models and examples on general multigraphs. To illustrate the practical usefulness of the algorithm, we will then apply this theory and the algorithm and predict hinge locations on several actual case study proteins. The selected proteins are classic hinge bending proteins with well annotated hinge locations available at the Molecular Motions Database MOLMOV [36], or in the literature.

Hinge prediction is a relatively mature area of research in bioinformatics and computationally biology, with several methods that can be applied [50, 51, 52], but our algorithm offers some unique features and can answer additional questions that were not previously considered. In short, we will illustrate with small examples on graphs and on actual proteins, that we are able to distinguish which linkers are just abstractly connecting the two rigid domains and which linkers are actually constraining their relative DOF, and should therefore be declared as actual hinges. This will allow us to give a more precise definition of hinge motions than those given in the literature. With the methods developed in this chapter, we can predict hinge locations to sub-residue accuracy, finding the actual bonds and atoms that are predicted to be part of the hinge. This will enable us to provide very specific and detailed predictions of hinge regions. In addition, our method is the first known Hinge-prediction algorithm to our best knowledge that can extract the number of degrees of freedom associated with the hinge motion (i.e. relative DOF between a pair of rigid clusters).

4.2 Hinge motions and background

Hinge bending motions in proteins is somewhat of a vaguely defined concept in the biological and computational biology literature. Hinge motions typically involve opening and closing motions between domains. These motions often control the access of ligands to the active site, which in many hinge proteins is located near the hinge region at the inter-domain interfaces [51]. A commonly used definition that will be initially useful for our purposes is that hinge motions occur between two well-defined rigid regions (often entire rigid domains) [52, 62, 81, 156]¹ connected by a flexible tether (linker) that move relative to each other.

Hinges in the literature can have multiple names, such as ‘domain linkers’, ‘bending regions’ [75], or ‘flexible tethers’ [10], which are basically flexible regions in a protein (a collection of rotatable bonds - which can be created by both covalent and non-covalent interactions [75]) that tether the protein rigid clusters and constrain their movement [81]. These mechanically important regions of the protein tend to occur on the surface of proteins rather than in the buried parts of the core, and since hinges are found in flexible regions of the protein, they mostly occur in loops and turns [51]². Many hinges act as dividers separating the domains, and provide a scaffold to prevent unfavorable interactions between domains [52, 81, 156]. They should be flexible in order to keep domains separate, and tight enough so that two domains can move with respect to each other in a constrained way [81].

Hinge motions typically occur between large rigid domains in the protein which can be composed of several secondary structures, but can also occur between smaller

¹We can think of a protein domain as a compactly folded region of a protein that has independent structural stability, and they can in some cases be composed of two, three hundred residues (amino acids), although in most cases they typically contain less than 100 residues [15] and in many cases a much smaller number. Domains are linked to other domains with flexible linkers and loops.

²Loops and turns are irregular and unstructured parts of the protein that lack secondary structure which often connect the regular secondary structure elements, such as α -helices and β -sheets and domains, see [15] for instance.

rigid regions, which are sometimes called fragment hinge motions. The residues on either side of the hinge are co-moving with their respective rigid clusters, and the two rigid clusters (domains) undergo relative rigid body movements about the hinge. Throughout the hinge motion there should be no significant steric clashes between the two clusters [51]. One mechanical analogy of hinge motions or domain closures is to think of the motion in the door, where the door opens and closes on the hinge. However, hinge motions are generally more complicated, for instance they may resemble general screw motions (combinations of translations and rotations).

One feature of hinge motions is that they can usually be typified by substantial changes in the backbone main-chain torsion angles (i.e. ϕ/ψ angles) of a few residues in a localized region (corresponding to hinge(s)) with little changes in the domains on either side of the hinge [51]. The small number of residues that undergo large main-chain angle changes represent the hinge location. This description is sensible as each larger rigid body (domain) on either side of hinge will move with a single rigid-body trivial motions (translations, rotations and their combinations) and should not have any internal motions.

To be consistent with the FIRST vocabulary, we will often refer to the two rigid regions/domains involved in the hinge motion as *rigid clusters*, or clusters in short, as in the rigid cluster decomposition FIRST terminology. The rigid clusters obtained in FIRST are always maximally rigid (i.e. maximally rigid subgraphs, see below).

It is also important to note that proteins can be composed of several discrete domains that are connected by inter-domain hinges, so a single protein can have more than one hinge.

In Figure 4.1 we have displayed the hinge in the protein inorganic pyrophosphatase, which is crystalized in both open (typically a ligand-free conformation of

the protein) and closed conformation. This protein represents a typical hinge motion, where the two domains connected by a single linker undergo a closure motion about the hinge which is located in the short loop region between the two domains (see below predictions on this protein). Like in Calmodulin (also shown later in the chapter), this example is perhaps the simplest hinge and consists of a single segment of the backbone separating the two subunits, also called a single-stranded hinge. However, in many hinge proteins it is also possible for the chain to pass multiple times between the two clusters (rigid domains) where the clusters are connected by multi-stranded linkers, or to have multiple independent hinge regions, as is the case in LAO Binding Protein and Adenylate Kinase, respectively (see section 4.5). It is reported [51] that most of the proteins with hinge motions have three or fewer hinge connections (hinge locations), with majority having one or two.

Hinge motions comprise the largest class of known protein motions as depicted in the Database of Macromolecular Motions (MOLMOV) [36, 51, 61] and represent about 45% of all motions in a representative set found in the MOLMOV database. Colloquially, all motions that occur with small number of degrees of freedom between a pair of rigid clusters are called hinge motions. The MOLMOV database classifies the domain motions and other protein motions according to the size of the mobile units and their packing. This database attempts to provide a detailed taxonomy of domain motions, and in addition to the general hinge bending motion classification, it includes shear motions (small sliding (translational) motions of two domains that preserve their well-packed interface), fragment motions, subunit motions which are typically allosteric motions (see next Chapter) and other motions (for example elbow motion in the FAB arm of immunoglobulin, see below) [61, 108].

Predicting hinge locations is an important and active area of research in computational biology, and various methods have been used to predict hinges. Some of

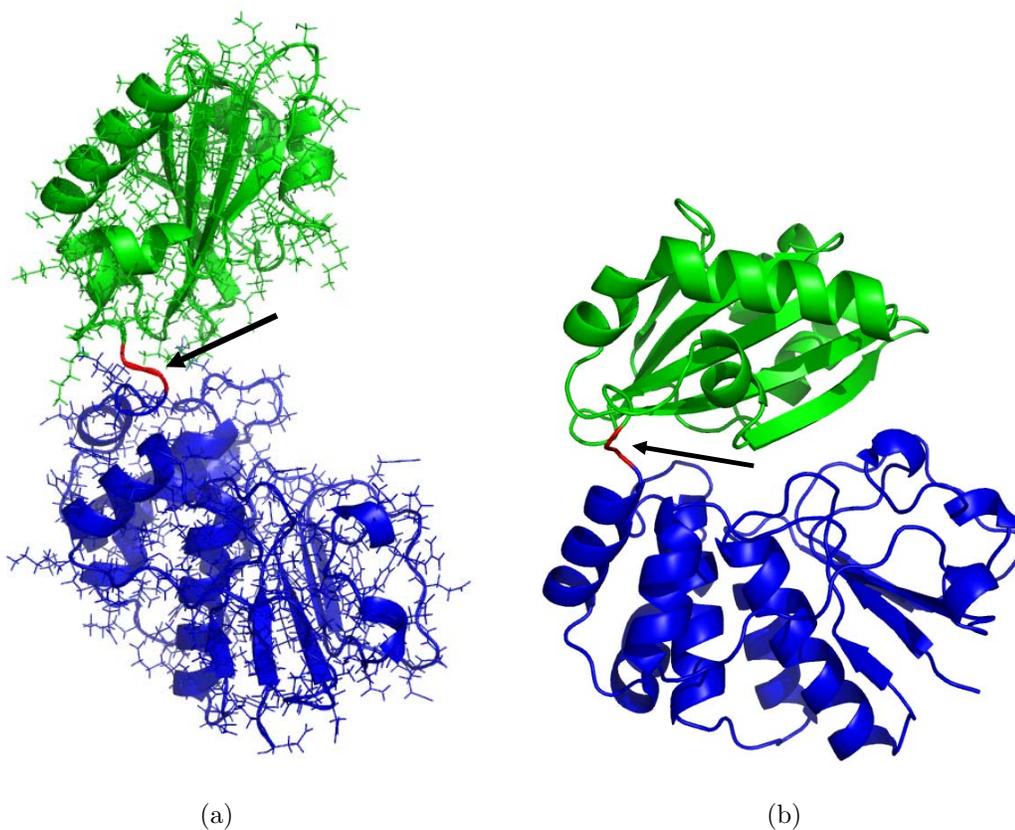


Figure 4.1: A typical hinge motion shown here on inorganic pyrophosphatase where the two domains undergo a closure motion. The hinge is indicated in red (location of hinge taken from Hinge Atlas Gold (HAG) standard dataset at Molecular Motion Database MOLMOV [36] with all atoms and bonds on domains on either side of hinge shown in green and blue. In (a) the open conformation (pbd id: 1k23) is shown in the cartoon representation with side-chains depicted with sticks and in (b) the closed state (pdb id: 1k20) is shown. Structures generated with Pymol.

these methods are based on sequence hinge-prediction, normal mode analysis techniques with the simplified Gaussian Network Model (GNM) (connectivity model) such as Hingeprot, energy calculation hinge predictions with program FlexOracle, predictors using two or more structures, morphing servers and predictors relying on combined methods [40, 49, 50, 51, 52, 76, 98, 156, 167].

As with most problems in bioinformatics, it is not surprising to see some attempts to predict hinges using only sequence (amino acid) information. Finding

hinges in this case is in some ways related to the problem of predicting domain boundaries from sequences and is sometimes tackled with standard techniques such as multiple sequence alignment [50, 51]. This is an important problem since a majority of proteins only have solved sequences and unknown 3-dimensional structure. However, as is the case with protein folding problem this is mostly considered an open problem [51].

Perhaps the simplest hinges to predict occur when the structure of the hinge protein has been solved in two or more different conformations (ideally both open and closed states) [156]. In such proteins it is sometimes possible to visually inspect the two states and (approximately) locate the hinge [36, 50, 51, 52], which can also be automated with methods such as Flexprot, Hingefind, and Dyndom [51, 76, 156, 167]. However, even when both structures are solved, some problems are encountered in precisely predicting the hinge, as the available open and closed structures can sometimes vary significantly in size (i.e. they may have a different number of residues) due to the form of the protein crystallized.

A much harder problem, and probably the most important and with the widest applications, is to determine the hinge location of the protein from a single 3-dimensional structure. Predicting a hinge location using a single structure is particularly important as often proteins only have a single known solved structure. Single-structure hinge prediction is the problem that we will consider in this chapter, and the Hinge-prediction algorithm that we propose only requires a single ‘snapshot’ protein structure. For proteins that undergo hinge bending motions, being able to predict the location of the hinge from a single set of structural coordinates (pdb file), can lead to new insights into the possible protein movements. As we have mentioned in the beginning of the chapter, and will discuss in more detail in the rest of this chapter, our algorithm can answer some new questions about the hinge motions and

offers a new perspective on hinge predictions. We now turn to the description of ideas and methods that are used to develop this novel Hinge-prediction algorithm.

4.3 Hinge motion detection via Relevant Regions

A number of mutational experiments have indicated that mutating or altering the length of hinges ³ in hinge bending proteins can significantly impact the underlying hinge motion, which can have consequences on the protein rigidity, domain-domain orientation and overall protein function [81, 124]. The paper by Hayward [75], specifically tries to inspect the distribution of different lengths of hinges between the pair of domains in various different proteins. The majority of hinges reported are one, two or three residues long [40, 51, 75], and the most common is two residues. It is also possible (although more rare) for hinges to be longer as they sometimes contain α -helical inter-domain linkers which are thought to act as rigid spacers separating two domains [59] ⁴.

The length of hinges is directly related to the relative rigidity/flexibility and the number of degrees of freedom that exist in the hinge motion (i.e. how restricted is the motion of one rigid body relative to the other, see below). Intuitively we can say the longer the hinge, the more flexible it will be. Some hinge motions can be limited (tight, with less degrees of freedom) like on a door [75] (door opens and closes around a hinge), while other hinge motions can be less constrained (more flexible) as in the elbow or ball-and-socket type motions. In the literature all of these are still called hinge motions if the motion occurred between two rigid domains.

³Length here is taken to mean the number of residues in the connecting region between two domains.

⁴Having helices in a hinge is sensible in terms of the relevant regions between the two clusters (see below), since a typical isolated α -helix comes up as rigid in FIRST (with enough hydrogen bonds preserved, see [207]), so it will act just as a single rigid body (i.e. atom) in the hinge.

In addition to identifying the hinge location, quantifying the flexibility of hinge motions, more specifically the relative DOF between the pair of rigid clusters in the hinge motion, will provide us additional important insight and an indication of how tight or loose the hinge is. The relevant regions algorithm 3.3.3, with some modifications provided below, is ideally suited to predict both hinge motions and quantify their DOF (i.e. relative DOF between two rigid clusters). We first motivate these ideas, which will be followed by more precise definitions and mathematical characterizations in terms of the pebble game algorithm and extensions of the relevant regions detection algorithm.

We have informally introduced hinge motions, as motions of one rigid cluster relative to the other rigid cluster (where the two clusters are connected by a flexible tether), as is traditionally defined in the literature. This is certainly a very vague definition of hinge motions. Perhaps a clearer definition is given in [81], where the authors say that the flexible tethers (hinge) which connect the two rigid clusters are constraining their movement. Hinges should constrain the movements of one cluster with respect to the other, so that one cluster is not freely moving relative to the other cluster. So, before we proceed with the discussion on how we use the relevant regions for predicting hinges, we first need to provide a more detailed discussion and a precise rigidity-based definition of what we mean by a hinge motion.

We will call any connecting region between the pair of rigid clusters a *linker*. In some sense, we want to be able to know which linkers should be called hinges.

Let us consider a pair of rigid cluster which are connected by a linker region(s) between them. If the linker has substantial length, it is possible that this linker only abstractly connects the two rigid clusters and acts as a very ‘loose’ connection. This type of a linker is expected to not constrain the relative DOF of the two clusters (i.e. we could freeze one cluster and the other one will still have all of its six trivial

DOF). With such linkers, the motion (DOF) of one rigid cluster is not restricted by the other rigid clusters. In terms of the relevant regions vocabulary, such long linkers are expected to be irrelevant with respect to the two clusters (we take two clusters to be core, see below) and would not be identified as a hinge. On the other hand, a tight (most likely a shorter) linker is expected to restrict (constrain) the relative DOF (motions) of the two clusters, and would act as a relevant region of the two clusters. With such a linker, a rigid cluster is not freely moving relative to the other cluster (below we will define their motions as being coordinated).

Loosely speaking (see below for more precise definition), we want to identify those linkers that are constraining (restricting) the motions of the two rigid clusters and call them *relevant linkers* and neglect all other *irrelevant linkers*. The relevant linkers would form the hinge locations. Relevant linkers are contained in the relevant region of the two combined rigid clusters.

In Figure 4.2 we have visually represented this discussion and some possible scenarios involving two abstractly connected rigid clusters. The gray linkers between the pair of rigid clusters (Figure 4.2 (a)) represent the irrelevant (overly flexible) linkers between the two clusters which have no impact on relative DOF between the two clusters. Similarly any flexible loop attached to either cluster will also have no impact. If the irrelevant linkers or other loops are cut (Figure 4.2 (b)) where we create a collection of loose dangling ends or completely removed (Figure 4.2 (c)) the relative DOF (motions) between the two clusters would remain unchanged. It is only the relevant linkers, shown in red, that can constrain the relative motions of the two clusters, and would be identified as hinges. In Figure 4.2 (d) both of the linkers are only abstractly connecting the two clusters, and have no impact on their relative DOF. In this case, we will say that there is no hinge motion, as all connections are irrelevant linkers. Each cluster has its own six (trivial rigid body) DOF and its

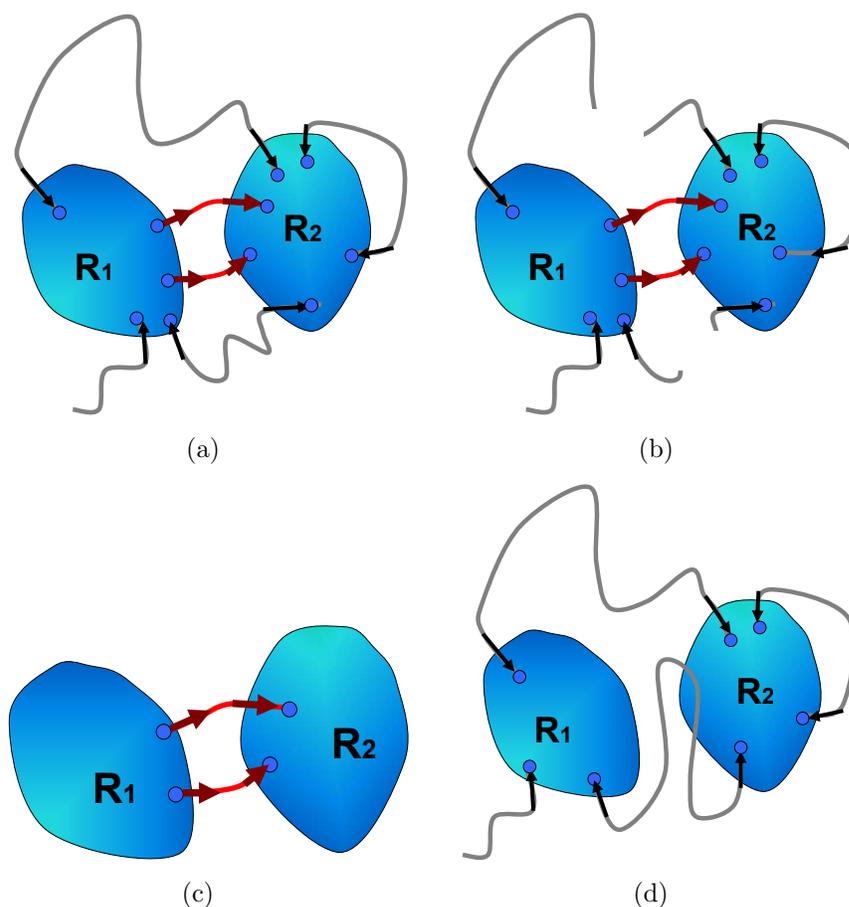


Figure 4.2: Possible scenarios that can occur in relative motion of two clusters as in hinge-bending motions or general domain-domain motions. Some linkers (shown in gray) between the two clusters just abstractly connect the two clusters in a graph sense and have no impact on the relative DOF of the two clusters, while other tighter linkers which are called relevant linkers (shown in red) are actually restricting the motions of one cluster relative to the other clusters and should correspond to the true hinges (a). As the irrelevant linkers and other loop-like attachments do not constrain the relative motion of the two clusters, they can be cut (leaving a collection of dangling ends) as shown in (b) or completely removed (c) and the relative DOF of the two clusters will be the same as those in (a). In addition to determining the location of hinges, this can be useful when trying to speed-up computations (i.e. MD simulations), where the focus of the simulations should be on the two clusters with the relevant connections, and it can assist in mutations studies, where only mutations in the relevant linkers could contribute to the change in the hinge motions. In (d) there are no relevant linkers (all arrows are directed into clusters), so we would say there is no hinge motion. The arrows indicate directions generated by the pebble game, that can be used to find the relevant regions. Directed paths from one cluster to another will be relevant linkers.

motions are independent of the motions of the other cluster (i.e. freeze or ground one cluster and the other cluster still retains its six DOF). The arrows in these examples represent the imagined output of the pebble game algorithm and is directly used to identify relevant (hinge) versus irrelevant (non-hinge) linkers (see below).

All of this can be precisely formulated and relevant linkers can be extracted with modifications of the relevant regions detection algorithm. We now state these ideas more precisely.

Let the two maximal rigid clusters in a suspected hinge motion that we want to test be denoted as R_1 and R_2 . The rigid clusters R_1 and R_2 are basically rigid (nonempty) induced disjoint subgraphs in the molecular multigraph G (see remark below why they have to be disjoint). We want R_1 and R_2 to be maximal rigid subgraphs, as it is not valuable to separate parts of a rigid cluster since all vertices (atoms) belonging to any rigid cluster will move rigidly together. Moreover, in the rigid cluster decomposition in FIRST, each identified rigid cluster is a maximally rigid subgraph, which can be identified with failed searches in the pebble game algorithm (see [84, 115]).

Remark. Note that R_1 and R_2 have to be (vertex) disjoint subgraphs ($R_1 \cap R_2 = \emptyset$), since in body-bar (body-hinge) frameworks two rigid clusters that share a vertex (body, atom) would have to form a larger rigid cluster, so they would not be maximal. An easy way to see this is to condense one of the rigid clusters into a single vertex, then the remaining structure (which includes this condensed cluster) basically consists of the other cluster which is rigid (see [115] for instance, for more formal proof). Each atom (vertex) has to belong to a unique rigid cluster. ■

Assume that the pebble game has been played on the entire graph G . Each rigid cluster, R_1 and R_2 , has six (trivial) DOF. We will combine R_1 and R_2 into core G_c (i.e. $G_c = R_1 \cup R_2$). The first step we need to perform is to draw maximum

number of free pebbles to the core (i.e. maximum free pebbles simultaneously to R_1 and R_2). This number will range from 7 to 12. It cannot be 6 since we assumed that R_1 and R_2 are two separate rigid clusters (if they were the same rigid cluster then clearly we would have 6 free pebbles on the core). Recall, that the maximum number of free pebbles we recover on the core quantifies the relative DOF (the possible motions) of the core. As we have done many times before, it helps to ignore the 6 DOF corresponding to trivial rigid body motions, and only focus on the remaining (internal) relative DOF of the core, which ranges from 1 to 6. We will sometimes call this number the *relative DOF between the two rigid clusters*, or *the DOF in the hinge*. Another way to think of the relative DOF between two rigid clusters, is to freeze or ground one of the rigid clusters, which essentially ignores the 6 trivial DOF, and check how many DOF are remaining in the other cluster.

6 relative DOF between two rigid clusters can theoretically occur due to many different possibilities, and in terms of amplitude or range of finite motion, it can encompass anything from a perfect bridge of length 6 (i.e. six single bonds (hinges) with 5 edges for each bond between the rigid clusters, see below for further definition) or longer bridges, up to a complete disconnection. Having 6 relative DOF between the two clusters (i.e. 12 free pebbles on core) means that we are able to simultaneously recover 6 free pebbles to one rigid cluster and 6 free pebbles to the other rigid cluster. So freezing (grounding) one rigid cluster has no effect on the initial motions (DOF) of the other cluster (one cluster is freely moving relative to the other cluster). In such a scenario, any potential linker that is connecting the two clusters would be identified as irrelevant and we would say that there is no hinge motion. Put another way, when the answer is 6 relative DOF between the two clusters, in terms of the relevant region detection algorithm, once we have drawn back maximum free pebbles to both rigid

clusters (i.e. core), there will be no outgoing edges out of either cluster. So, there will be no relevant linkers. We can state this basic observation as a proposition:

Proposition 4.3.1 *Given a molecular multigraph G . Let R_1 and R_2 be two distinct rigid clusters (maximal rigid subgraphs), and let core $G_c = R_1 \cup R_2$. Assume the pebble game is played on G and maximum number of free pebbles are drawn to vertices in G_c , and the number of free pebbles recovered is 12. Then, (in the pebble game generated directed graph) there are no outgoing edges out of vertices in G_c .*

Proof Since maximum number of free pebbles on the vertex set of G_c is 12, and R_1 and R_2 are separate rigid clusters, there has to be exactly 6 free pebbles in R_1 and 6 free pebbles in R_2 (recall we can always recover six free pebbles to any vertex and exactly 6 free pebbles to any rigid cluster - Lemma 3.2.3). Assume in the pebble game generated directed graph there is an outgoing edge from one cluster, then this outgoing edge and the corresponding failed search region would further increase the size of the rigid cluster, but this cannot happen since the clusters are already maximal rigid subgraphs. ■

The more interesting cases clearly occur when the number of relative DOF between the two clusters is between 1 and 5 (i.e. 7 to 11 free pebbles in G_c). In this case the motions of one cluster are restricted with respect to the other cluster.

We can now state more precisely what we mean by a hinge motion:

Definition 4.3.2 *Given a molecular multigraph G . Let R_1 and R_2 be two distinct rigid clusters (maximal rigid subgraphs), and let core $G_c = R_1 \cup R_2$. Assume the pebble game is played on G and maximum number of free pebbles are drawn to vertices in G_c . When the number of free pebbles on G_c is less than 12 (i.e. relative DOF between R_1 and R_2 is between 1 and 5), we say there exists a hinge motion, otherwise we declare there is no hinge motion (number of free pebbles on G_c is 12).*

When there is less than 12 maximum free pebbles at $G_c = R_1 \cup R_2$, we will say that the two rigid clusters R_1 and R_2 are in a *coordinated motion*, and that their *movements are coordinated*.

We chose the word coordinated, as the motions of one cluster are restricted and depend on (are coordinated with) the motions of the other cluster, in the sense that freezing one cluster reduces the possible motions (DOF) of the other cluster.

Note that when we declare ‘there is a hinge motion’ (i.e. two clusters are in a coordinated motion) or ‘there is no hinge motion’ between two rigid clusters R_1 and R_2 , the answer will always be the same regardless of the play of the pebble game (i.e. which order the edges are tested in the pebble game algorithm). In other words, the count of relative DOF between the two clusters (free pebbles reversed to the core take away 6) is not dependant on the play of the pebble game. This follows since we know that the maximum number of pebbles we can recover to any subgraph is invariant under the plays of the pebble game (Theorem 3.3.2).

When the maximum number of free pebbles on the core is less than 12, not only does this tell us that a hinge motion is possible, but the relative DOF number can give us some further insight. For instance, a hinge motion with say one or two DOF is more ‘tight’ and would have a more of a constraining (restrictive) effect on the motions of the two rigid clusters than would a hinge motion with four or five DOF. We will revisit this more in the later sections when we apply the algorithm to protein hinge motions.

Using our definition of a hinge motion, as a mental image we freeze (ground) one of the rigid clusters (or equivalently freeze six pebbles on its vertices) say R_1 for instance, and see how many free pebbles can be recovered to R_2 without touching the free pebbles on R_1 . Having a hinge motion between R_1 and R_2 means when pebbles on R_1 are frozen, then there is a permanent reduction in the maximum number of

free pebbles (DOF) that can be recovered to the other rigid cluster R_2 (i.e. less than 6 free pebbles). This permanent reduction is marked by an outgoing edge out of core at R_2 , once maximum free pebbles are recovered. Another way to view this is as follows: having six pebbles frozen at R_1 , and only 1 to 5 available maximum free pebbles at R_2 means we would have to add only 1 to 5 edges (bars), respectively, between R_1 and R_2 (one edge for each extra free pebble) to rigidify them into a single larger rigid component. On the other hand, for any two rigid clusters that are just abstractly connected in a graph sense (or not connected at all) and not involved in a hinge motion, a full six edges (bars) need to be added between the two clusters to rigidify them into a single cluster.

A further and very important consequence of having 1 to 5 relative DOF between R_1 and R_2 or a coordinated motion, is that there must exist a relevant linker connecting the two clusters that is constraining their relative motions, which we use to identify a hinge location. As there is less than 12 free pebbles we can simultaneously get to R_1 and R_2 , a loss of at least one free pebble means that a linker connecting R_1 and R_2 must have permanently removed some free pebble(s), which on the pebble game generated directed graph is captured by an outgoing edge out of the core. In terms of the above discussion, once we have recovered maximum free pebbles to the core (R_1 and R_2 combined) the relevant region (the capped failed search region) will be the *relevant linker*. Such relevant linker(s) cause two clusters to be in a coordinated (i.e. restricted) motions, and will be the predicted hinge location (see also Algorithm 4.4.1 and Figure 4.4). This allows us to give a very detailed definition and specification of a hinge location. Another helpful way to think of a hinge location (relevant linkers) between the two rigid clusters, is that a removal of edges in the hinge will result in the changed number of DOF between the two clusters, whereas removal of edges in any (non-hinge) irrelevant linker that is just abstractly connecting the two

clusters in the graph sense will leave the same DOF between the two clusters. This approach, with the extensions of the relevant region detection algorithm, will be used to predict hinges in proteins in the later section.

Proposition 4.3.3 *Given a molecular multigraph G . Let R_1 and R_2 be two distinct rigid clusters (maximal rigid subgraphs), and let core $G_c = R_1 \cup R_2$. Assume the pebble game is played on G and the maximum number of free pebbles are drawn to vertices in G_c , and the number of free pebbles recovered is less than 12 (i.e. 7 to 11 free pebbles). Then there exists a relevant region connecting R_1 and R_2 .*

Proof Assume that when we recovered the maximum free pebbles to the core G_c that six free pebbles are placed (and frozen) in one of the clusters, say R_1 (on any of its vertices). As R_1 is maximally rigid it will not have any outgoing edges. On the vertex set of the other cluster, R_2 , there must be $6 - m$ free pebbles, where m is an integer between 1 and 5.

In the pebble game generated directed graph, R_2 will have exactly m outgoing edges ⁵ (Lemma 3.2.3) ⁶. A single outgoing edge out of R_2 will guarantee that there will be a failed (capped) search region, where the directed path search for the free pebbles from R_2 must reach R_1 and subsequently the vertices holding the free pebble(s) in rigid cluster R_1 . It is clear that in a failed search for the free pebble(s) from R_2 that we will never be able to reach a free pebble at any vertex except those belonging to R_1 , otherwise that would increase the total number of free pebbles in the core G_c , so there must be a relevant region between R_1 and R_2 , ■

Note that if we alter the relevant linker, for instance by removing an edge, the relative DOF count between R_1 and R_2 would be changed, and hence the underlying

⁵Note that no matter how we recover the $12 - m$ free pebbles or have played the pebble game, the core will always have m outgoing edges.

⁶These outgoing edges can be used to search for the m free pebbles from R_2 (one free pebble per each outgoing edge giving m edge-disjoint directed paths from R_2 to R_1), which leads to a relevant region.

nature of the hinge motion would be different. On the other hand, increasing the size of irrelevant linkers, for instance, should not impact the hinge motion. The effect of these changes is easily testable with our methods and algorithms. This may have important consequences and biological applications such as mutation studies, where it is known that modifying the length of hinges by mutations can impact the function of hinge-bending proteins [81]. This will be discussed more later in the chapter.

As an illustration of these ideas, in Figure 4.3 we have shown some simple possible scenarios of hinge motions and when there would be no hinge motion between the two rigid clusters, using simple linker connections, which we call a *perfect bridge*.

We define a *perfect bridge* between two rigid clusters (which can be represented by a pair of vertices, one vertex for each rigid cluster) as a path connecting the two clusters where all the bonds in the path are single (rotatable) bonds (i.e. in the multigraph they are identified with 5 bars between its end vertices). We define the *length* of a perfect bridge as the number of single bonds in the bridge. Using a simple count of constraints imposed by bonds (i.e. 5 bars for each bond) and 6 DOF introduced by each vertex (atom), we can quantify the effect of a perfect bridge on the number of DOF. If the perfect bridge has n atoms ($n \geq 0$) (excluding any vertices from each cluster) and $n + 1$ single bonds (which is its length) between the pair of clusters, then the contribution of the perfect bridge to the overall DOF count will be $6 \times n - 5 \times (n + 1) = n - 5$.⁷ A double bond essentially locks its two bonding atoms into a single rigid body, so a presence of a double bond in the bridge would basically shorten the length of a bridge by one. From this count, a perfect bridge of length 1 (i.e. $n = 0$, a single bond between the rigid clusters) will remove 5 DOF leaving only

⁷Another way to see this is to start with one rigid cluster, say R_1 , attach a one valent cluster (vertex) to it with 5 bars, which introduces one additional DOF. We then attach a one-valent vertex (rigid cluster) to the new introduced vertex, which introduces yet another DOF. We keep building up the chain, and once we have built a perfect bridge of length 6 (or $n = 5$ vertices (atoms) in the bridge) between the first and last rigid cluster there will be a total of 12 DOF ($6 + 6$) DOF.

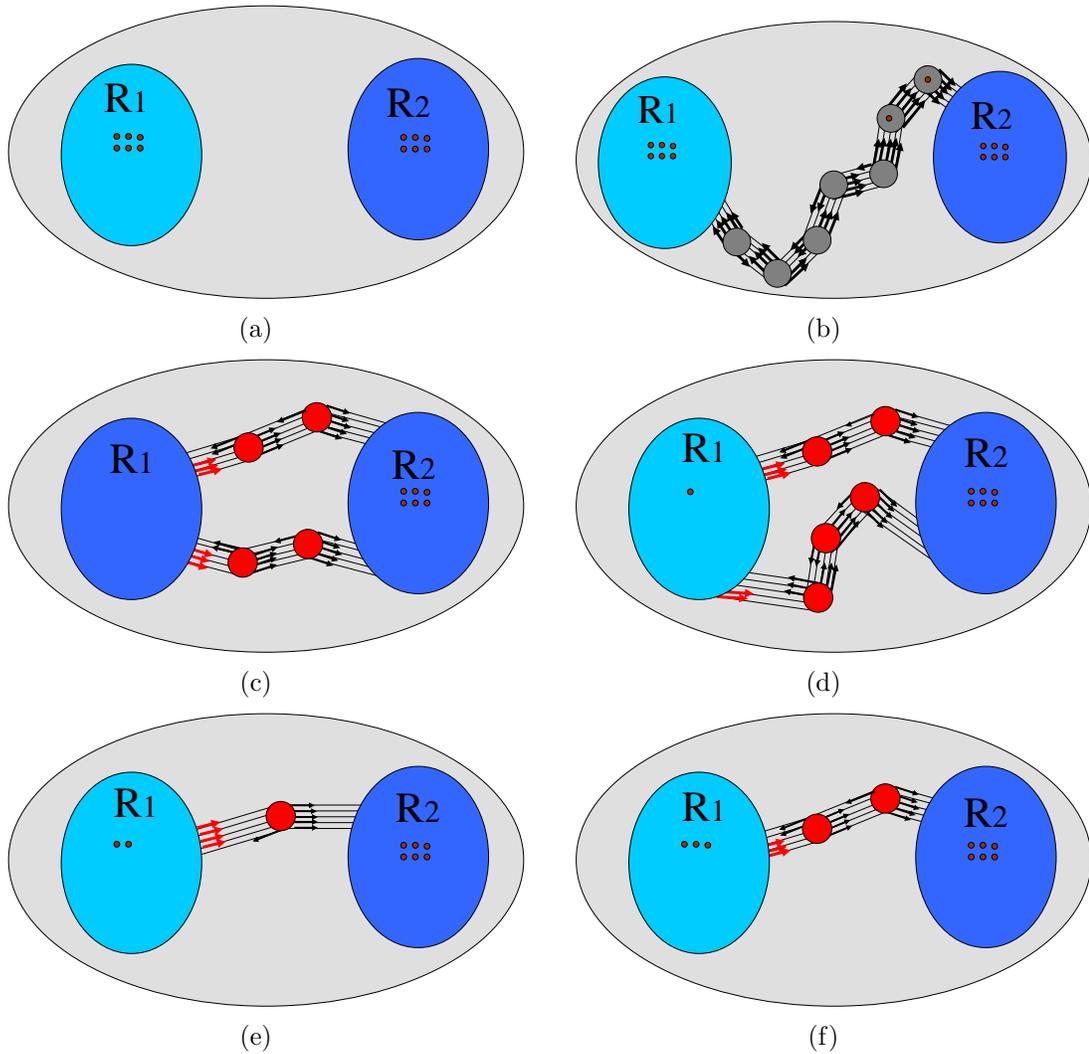


Figure 4.3: *Simple hinge and non-hinge linkers and the extraction of DOF in the hinge from the pebble game.* In (a) rigid clusters R_1 and R_2 are disconnected, each having independent 6 trivial DOF. In (b) the two clusters are connected by a perfect bridge of length 8, which is too long to restrict the motions of one cluster relative to the other. Both (a) and (b) have 12 free pebbles remaining on R_1 and R_2 , and even though (b) has a linker, in both cases we indicate there is no hinge. In (c) there is no hinge motion as R_1 and R_2 are locked by two perfect bridges of length 3 into a single rigid cluster. The scenarios in (d), (e) and (f) are hinge motions of 1, 2 and 3 DOF, respectively (i.e. two rigid clusters are in coordinated motions). The vertices in the relevant linkers are coloured in red. Note (in the relevant linkers) the existence of a directed path(s) from one cluster to another (starting with red outgoing arrows from R_1), indicates the relevant linker (hinge) is constraining their motions.

one DOF ($(6+6) - 5) - 6$ trivial DOF = 1) between the clusters, a bridge of length 2 will remove 4 DOF leaving two non-trivial DOF, a bridge of length 3 will remove 3 DOF, a bridge of length 4 will remove 2 DOF and a bridge of length 5 will remove only 1 DOF.

Note that any perfect bridge of length less than 6 will be a relevant linker between the two rigid clusters. On the other hand, a perfect bridge of length 6 or bigger will not be constraining the motion (i.e. irrelevant linker) of the two clusters as each cluster will independently have its 6 trivial DOF where no DOF are being constrained by the other cluster. In fact, bridges longer than 6 will start to introduce additional DOF to the overall graph.

Referring to Figure 4.3, in (a) the two rigid clusters are completely disconnected subgraphs⁸, and in (b) there is a perfect bridge of length 8 connecting the two clusters. In both of these cases, both rigid clusters simultaneously retain 6 free pebbles (i.e. 12 free pebbles recovered on the core) so we declare that there is no hinge motion. Even though (b) has a linker between the two clusters, however since this linker does not remove any free pebbles from R_1 or R_2 (i.e. they are not in a coordinated motion), we say that this linker is irrelevant. In (c) there are two perfect bridges of length 3 between the clusters. From discussion above, a perfect bridge of length 3 will remove 3 DOF, so two of these bridges will lock the two rigid clusters into one larger rigid cluster (identified with only 6 remaining free pebbles) where both clusters will move as a single rigid body, so again there is no hinge motion. In (d), (e) and (f) we have an example of hinge motions with 1, 2 and 3 DOF, respectively (visually seen by the number of remaining free pebbles – 6). In (c) there are two perfect bridges, one bridge is of length 3 (removes 3 DOF) and one bridge of length 4 (removes 2 DOF) which leaves 1 relative DOF between the pair of rigid clusters.

⁸If one thinks of proteins, it is of course unrealistic to think of the two clusters as completely disconnected, but nevertheless, this serves a good motivational purpose.

Such hinges with simple bridges between the clusters can occur in real proteins but of course the graph generated from the protein structure could contain a lot more complex connections. Hinge regions in proteins could have multiple strands (linkers) between the two rigid clusters, and the linkers themselves could also contain rigid clusters bigger than a single atom, for example small rigid rings or even short helices. The hinge regions could also have non-covalent constraints (hydrogen bonds or hydrophobic tethers) [75] either within the hinge residues of the same linker or if there is more than one linker, non-covalent interactions between the different linker (hinge) residues are also possible. Non-covalent interactions can also occur between the hinge residues and the residues of the two large rigid clusters (see below). As the relevant region detection algorithm is applied on the general graph, it could equally easily differentiate relevant versus irrelevant linkers or any of these much more complex hinge linkers just discussed. We now turn to applications of these ideas to predicting hinge motions in proteins.

Remark. Previous attempts to use the rigidity based techniques to predict hinges (method Stonehinge [52, 98]) were significantly limited. These methods have not considered the careful type of analysis of relevant and irrelevant constraints that we do with the pebble game algorithm in this work, and as a result they would often report very long lengths of hinges. The method Stonehinge which uses an earlier implementation of a version of FIRST called PROFLEX [142], essentially identifies a hinge whenever there is a connection (linker) between a pair of rigid clusters. So, in this method, all the residues along the backbone connecting the pair of rigid clusters are identified as hinges. The obvious limitation of this approach is that it will lead to an an identification of overly flexible inter-domain linkers as possible hinges. It is clearly excessive to assume that all inter-domain linkers should correspond to hinges. The authors of this method report that this method leads to many false positive

predictions which in many cases is caused by a significant overestimate of the length of hinges [52, 98].

In terms of our vocabulary, the problem with the Stonehinge approach is that it does not distinguish relevant from irrelevant linkers between the pair of clusters. Stonehinge would often report both irrelevant and relevant connections as potential hinge locations, and by declaring irrelevant linkers as hinges, it would declare hinges which span more than ten, twenty and in some cases more than thirty residues [52]. Clearly such long flexible linkers will not constrain the motions between a pair of clusters, and they could not possible lead to a coordinated motions. As we have seen with simple examples in this section (and in proteins below) such overly flexible linkers should be declared as irrelevant, and discarded as they would not correspond to the hinge locations.

It should not be surprising and coincidental that the evolution of majority of hinges in hinge-bending proteins has selected small lengths (one, two or three residues [51, 75]). Our adapted relevant region procedure as is utilized in the Hinge-prediction algorithm below, is designed to only declare hinges (relevant linkers) that are coordinating (constraining) the motions of the two clusters, and we anticipate that this will lead to very specific and precise prediction of hinges. ■

4.4 Hinge-Prediction algorithm for proteins

We build on the ideas and definitions given in the previous section and introduce a novel hinge-prediction algorithm, which we call the *Relevant Hinge-prediction Algorithm* or Hinge-prediction Algorithm in short. Roughly speaking, our Hinge-prediction algorithm starts with a protein structure (i.e. pdb file), performs some standard file preprocessing steps for FIRST analysis, obtains the rigid cluster decomposition from FIRST and chooses two appropriate (typically largest) rigid clusters (again denoted

as R_1 and R_2) for which there is a suspected hinge motion. The algorithm then finds the relevant region of the core $G_c = R_1 \cup R_2$, which will correspond to the predicted hinge location.

Algorithm 4.4.1 – Hinge-prediction algorithm:

Input: PDB file.

Output: Yes, hinge motion exists, no otherwise. If hinge motion exists provide hinge location and DOF of the hinge (i.e. relative DOF between the two rigid clusters in the hinge motion).

File preparation: For a given protein (PDB file) add missing hydrogens (particularly to all x-ray structure) (i.e. using WHATIF [195] or REDUCE [151] programs). Remove water molecules, ligands and other heteroatoms.

- (1.) Run FIRST and choose two large rigid clusters and set them as core G_c .⁹
- (2.) Apply the Relevant regions detection algorithm 3.3.3 and find the relevant region¹⁰ of the core (i.e. $G_R \setminus G_c$) and relative DOF of the core (i.e. pebbles reversed to the core).

If the maximum number of free pebbles recovered to the core is less than 12, then declare there is a hinge motion and the relevant region (atoms and bonds) of the core is the predicted hinge location, otherwise declare no hinge motion exists. If the number of free pebbles at core is < 12 , the DOF of the hinge (i.e. relative DOF between two clusters) is the maximum pebbles recovered to the core – 6.

⁹The easiest way to determine which two rigid clusters to choose is to run FIRST at hydrogen bond energy cutoff 0.0 kcal/mol and obtain a dilution plot (see [109] for details on dilution plots and also Chapter 6) and pick an energy cutoff corresponding to the two clusters (typically two largest clusters should be chosen, but any pair of rigid clusters could be studied).

¹⁰In the general implementation of the Relevant regions detection algorithm in the add-on to FIRST, in order to remove the ambiguity of the relevant regions, we grow the relevant region to an enlarged relevant region (see Chapter 3). Note that, this will never change the answer to whether there is a hinge motion between two rigid clusters, and the DOF count of the hinge.

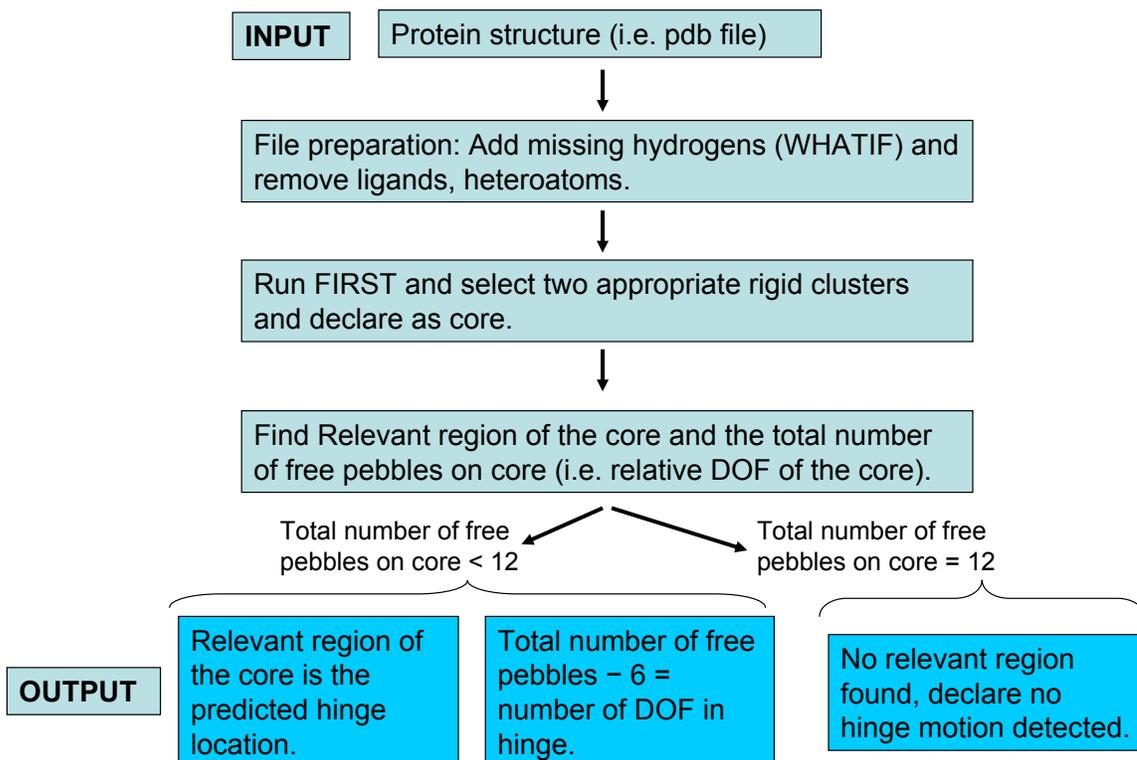


Figure 4.4: Outline of the Hinge-prediction algorithm 4.4.1. Starting with a single snapshot (pdb file) of the protein, a hinge is defined when there are less than twelve maximum free pebbles that can be retrieved simultaneously to the two rigid clusters. The output of the algorithm is the predicted hinge location (consisting of relevant atoms and bonds) and the count of DOF in the hinge (relative DOF between the two clusters).

In Figure 4.4 we have shown a schematic flow chart of the Hinge-prediction algorithm. Once we have selected the chosen protein to test out for the possible hinge location, prior to running the algorithm there are some standard file preparation steps that need to be performed in order to perform the FIRST analysis. Recall that we only need a single structure to run the Hinge-prediction algorithm. As hydrogens are typically not included in most structures solved by X-ray crystallography, we add

hydrogens with an external program WHATIF [195]. There are other comparable programs that can also be used, such as Reduce [151].

We will remove all the ligands (when present)¹¹ and water molecules from the structure and exclude them from the created constraint molecular multigraph G ¹². Removing these extra heteroatoms can sometimes slightly change the rigidity and flexibility of the protein. However, in our samples whether ligands were included or excluded from the structure, it had no effect on the predicted hinges in any of the proteins. As a further side comment, ligands could in some cases add more rigidity to the protein as they can form locking links between the domains (often temporary hydrogen bonds that are broken/formed during opening/closing hinge motions) and rigidify the two domains and cause the actual hinge region to no longer appear flexible in FIRST. Ligand-free structures should enable identification of the intrinsic rather than ligand-induced flexibility of the protein, which is why the open state (ligand free) protein (when structure is available) generally allows for more accurate predictions of hinges [98]. In the open state the domains are usually more separated in space, so hinges should be more easily identified, which is another reason open state structures are preferred [152]. In addition, it is often the case that only an apo-structure (ligand free) of the protein is solved by crystallography and the ligand-bound structure is not known.

The first step in the algorithm is to run the FIRST analysis on the chosen protein and obtain the rigid cluster decomposition. Once the rigid cluster decomposition is obtained, we select two large rigid clusters as the possible clusters involved in the hinge motion. We generally recommend choosing the two largest rigid clusters if they

¹¹Note that the artificial removal of the ligands from the pdb structure (if the structure came with ligands) still keeps the rest of the protein in the ligand-bound conformation.

¹²If one desires, completely buried water molecules can be kept with the program PRO_ACT [209] as the protein can form hydrogen bonds with the water molecules. However, the authors of [122] suggest that water molecules can be removed prior to FIRST analysis.

are in close proximity to each other. The most efficient way to determine which two rigid clusters to choose is by obtaining a hydrogen bond dilution plot (see Chapter 1) and also Figure 4.10. We then pick a hydrogen bond energy cutoff corresponding to the two clusters. Alternatively, when there is ambiguity about which two clusters are the two biggest clusters (for instance the second and third rigid clusters are roughly the same size and contain more or less the same number of residues), more than one pair could be selected and the Hinge-prediction algorithm could be applied more than once (see the Adenylate kinase example below). Smaller rigid clusters could be selected but these are more likely to be defined in the literature as fragment hinge motions rather than hinge-bending domain motions.

Once the two rigid clusters are selected, we then declare them both as core, and find the relevant region. If the two rigid clusters (core) have less than 12 maximum recovered free pebbles, then there exists a relevant region between them and we declare this relevant region as the predicted hinge. When there is a hinge motion between the two clusters, the Hinge-prediction algorithm outputs the hinge location (atoms and bonds forming the relevant region) and the relative DOF between the two cluster. Otherwise, the core has 12 free pebbles and we declare that there is no hinge motion.

We should also mention that if the Hinge-prediction algorithm is run on the protein solved by an NMR file (or whenever ensemble structural data is available), we suggest that the FIRST rigid cluster decomposition be obtained using our FIRST-ensemble algorithm given in Chapter 6.

4.5 Application of hinge-prediction algorithm on proteins

We now apply our Hinge-prediction algorithm on case study proteins from the Hinge Atlas Gold (HAG) dataset available from the Macromolecular Motions database server [36, 49, 51, 52, 61, 108]. This dataset contains manually annotated hinge locations of some well known hinge bending proteins, obtained from literature based on experimental methods and in some cases by visual inspection when the structure is solved in both open and closed conformations. The case study proteins we chose to test out our algorithm are classic hinge bending proteins and were selected due to their well known hinge locations, both in HAG and literature, making them easy for comparison. Furthermore, these proteins represent a broad range of classes of hinge motion, which should illustrate the wide applicability of the Hinge-prediction algorithm. We also select other proteins that are not listed in HAG but appear in the literature.

Some hinges will be simple and consist of a single connection along the backbone (main-chain) of the protein separating the two rigid clusters (domains). Other hinge motions will be more complicated and involve several hinges passing multiple times between the rigid clusters. One protein particularly is picked because it has both long and overly flexible linkers and some shorter linkers, as we wanted to test out if our algorithm would correctly distinguish between the long flexible linkers which we suspect should be irrelevant and correctly identify other relevant hinges. Another protein was chosen as it is composed of several rigid domain regions, which has two independent hinge-bending motions between two different pairs of rigid clusters (domains) and no hinge motions between other pairs of rigid clusters. We will also apply

our algorithm on some proteins that are not annotated in HAG, but for which evidence of hinges and domain-domain motions exists in the literature. Additionally, we chose to test out our algorithm on another protein that is not listed as a classical hinge-bending motion protein, but is known to undergo relative domain-domain motion.

A majority of our predictions will be given for hinges (relevant regions) that occur along the back-bone (main-chain) of the protein, and the hinge predictions will be given in terms of residue numbers. We chose this option so we can make realistic comparisons of our predictions with the known hinge locations in HAG and literature, which only report residue numbers. However, as we will soon see, our method can give detailed hinge predictions to the sub-residue accuracy (atoms and bonds), as we can also detect any side-chain interactions (hydrogen bonds, hydrophobic interactions) that are part of the relevant regions and constrain the possible relative motions of the two clusters. These additional constraints can be crucial as their presence can cause coordination (i.e. coordinated motions) of the two clusters in the hinge. This unique feature of our algorithm will be specifically addressed on couple of proteins.

Inorganic pyrophosphatase

In Figure 4.5 (a) we have shown the rigid cluster decomposition of Inorganic pyrophosphatase (pdb id: 1k23) and the output of the Hinge-prediction Algorithm 4.4.1 (Figure 4.5 (b)). We see that majority of this protein is composed of two large rigid clusters. We use the same notation from the previous section to denote the rigid clusters, R_1 and R_2 , and the core region which combines these two clusters as G_c . When we combine the two rigid clusters, indicated with blue and green, we colour the combined core region with blue. We predict the hinge to be located in residues 180 and 190, and by careful inspection we see that the hinge forms a simple bridge

of length 3, which is a hinge of 3 DOF. This is confirmed by the Hinge-prediction algorithm, as we were able to get 9 free pebbles to the core ($9 - 6$ trivial DOF = 3 DOF). Identification of this hinge as a relevant linker between the two clusters (where all the atoms in the relevant region are shown as red spheres) gives an exact match of the hinge location with the HAG.

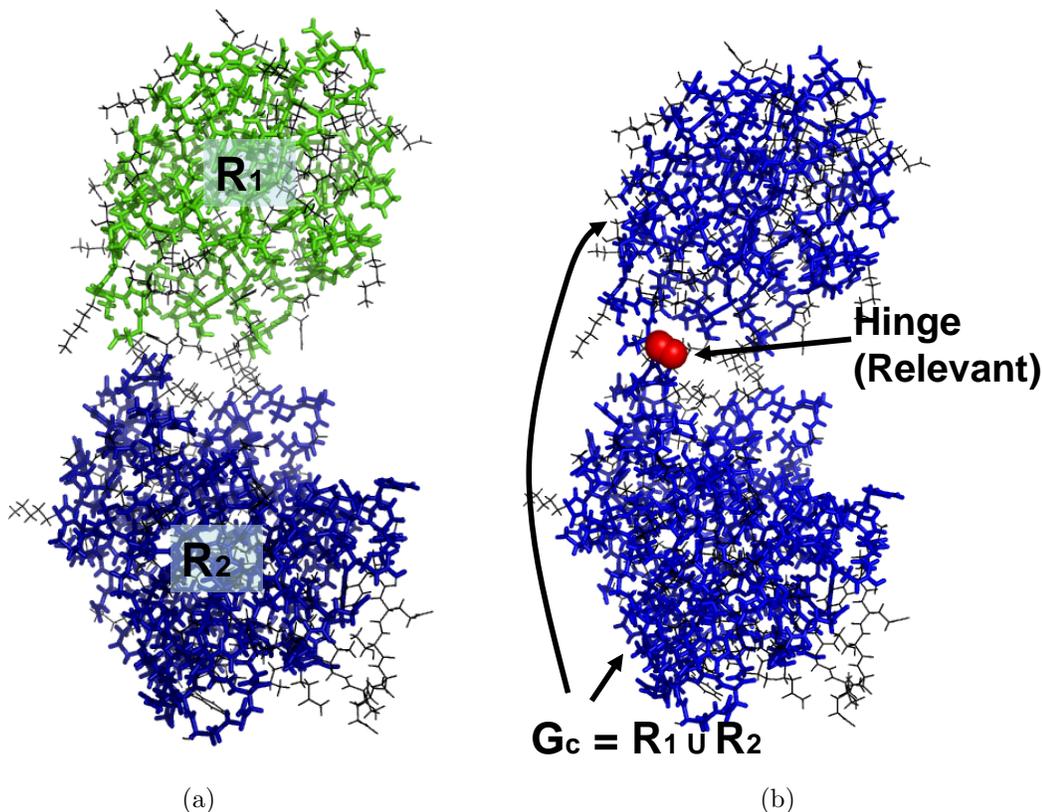


Figure 4.5: Hinge prediction of inorganic pyrophosphatase (pdb id: 1k23). In (a) the two rigid clusters R_1 and R_2 (blue and green) as obtained from FIRST rigid cluster decomposition are shown. The gray regions are flexible regions. Combining the two rigid clusters into core (blue), the Hinge-prediction algorithm 4.4.1 finds the hinge (b) to be in residues 89 and 90 (shown in red spheres), which exactly matches the HAG residue location. There are 3 relative DOF between the two clusters. Note the (gray) irrelevant regions that are just attached to the rigid clusters but do not contribute to the hinge motion.

In this protein there are two noteworthy strong hydrogen bonds as identified with FIRST that form between the two rigid clusters, one between side chain groups

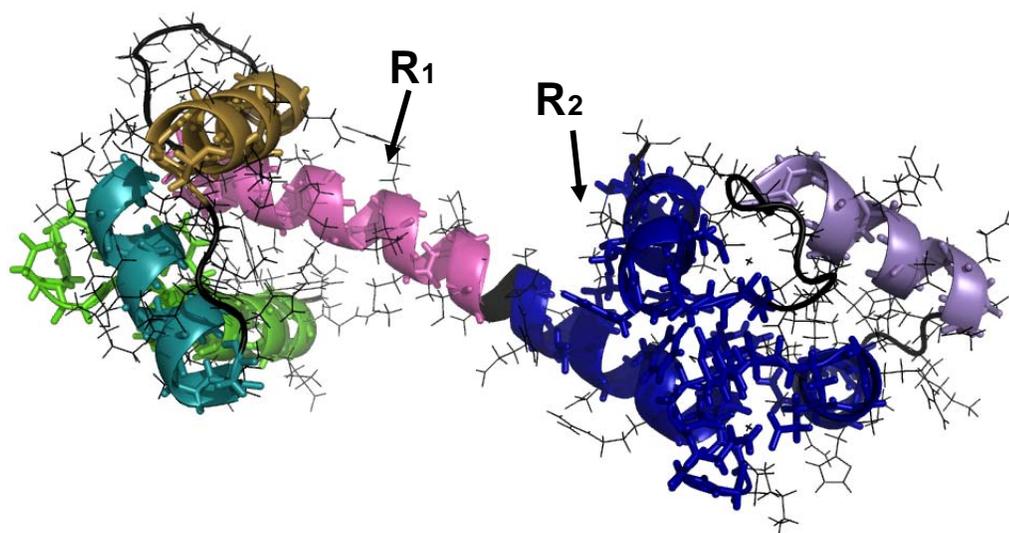
of Lysine 154 and Glutamic Acid 259 and the other one between Lysine 186 and Aspartic Acid 226. Both of these hydrogen bonds form very long linkers (bridges) between the two rigid clusters, and are correctly identified to be irrelevant with respect to the core (two rigid clusters). This is not explicitly highlighted in Figure 4.5, but these irrelevant linkers can be still visually seen as black thin sticks forming long flexible connections between the two clusters.

Calmodulin

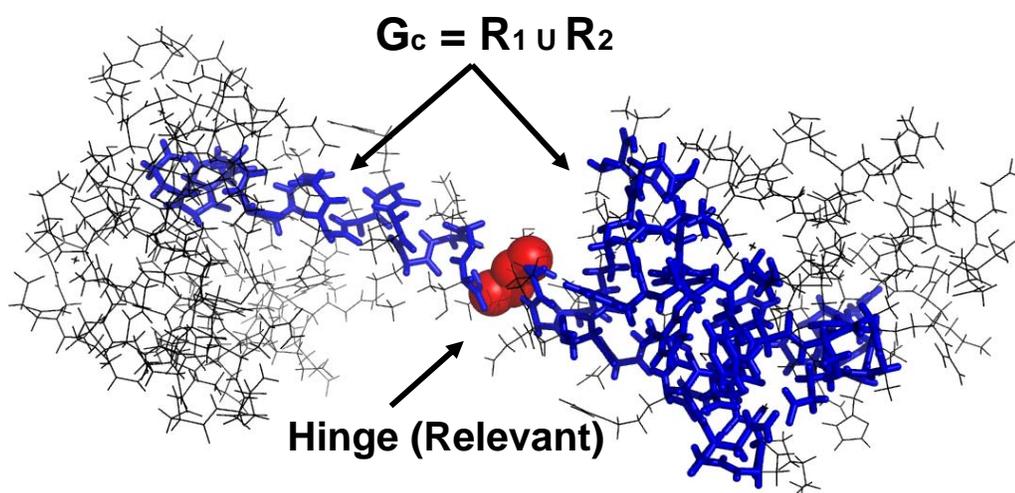
In a second test case, we have predicted the hinge location of calmodulin, nearly a 150 residue long, calcium binding protein. This highly conserved protein has somewhat of a atypical hinge as the hinge is not located in a loop or turn. Calmodulin has a dumbbell-shaped structure with a very long helix called the ‘central helix’ that is known to be flexible close to the middle of the central helix, allowing the protein to slightly unwind and undergo functionally important large-scale hinge bending motion [10, 193].

The rigid cluster decomposition of Calmodulin (calcium-bound open state) (pdb id: 3cln) is given in Figure 4.6 (a) and the corresponding hinge prediction in Figure 4.6 (b). Our Hinge-prediction algorithm finds residues 80 and 81 form a hinge (relevant linker). This prediction again gives an exact match to the HAG hinge. We find Calmodulin hinge motion to have only a single DOF (i.e. maximum 7 free pebbles are recovered to the two rigid clusters), which would mean that it has a highly constrained coordinated motion between the two clusters. ■

As we had briefly mentioned earlier, it is well known that non-covalent interactions can form parts of hinges and both main-chain and side-chain hydrogen-bonding interactions between the rigid clusters (domains) can be important factors in constraining and limiting the motions of one cluster relative to the other [75]. In fact,



(a)



(b)

Figure 4.6: Hinge prediction of calmodulin (pdb id: 3cln). Rigid cluster decomposition with two largest rigid clusters R_1 and R_2 forming a long central helix (a). The distinct colours represent the rigid clusters, with black/gray regions corresponding to the flexible parts of the protein. Our prediction of the hinge (shown in red spheres) between the two large rigid clusters is residues 80 and 81 matching exactly the HAG hinges. This hinge has 1 DOF.

some proteins can form strictly non-covalent interactions that form hinges between

residues remote along the sequence [75], which form relevant linkers and yet the backbone linkers would not form hinges (see also next example).

Calmodulin Continued: Non-covalent hinge forming interactions are also detectable

In calmodulin we also find that there is a relevant region formed by the hydrogen bonding interactions between the two rigid clusters (see Figure 4.7). A hydrogen bond between the backbone Oxygen of Asp78 and backbone hydrogen of Glu 82 forms a relevant linker and is constraining the motion of the two clusters. This linker is essentially a perfect bridge of length 3, so it contributes to a removal of 3 DOF between the two clusters. As this is a hinge motion of 1 DOF, the other 2 DOF are removed by the predicted hinge location along the backbone of residues 80 and 81. These two residues essentially form a bridge of length 4, which removes 2 DOF between the two clusters. The net effect on the DOF of the main-chain hinge linker and the non-covalent linker is then a removal of 5 DOF ($-2 - 3 = -5$), leaving 1 relative DOF between the rigid clusters.

In the central flexible hinge region of calmodulin, it is clear that some backbone N-H to C-O hydrogen bonds have to be weakened to disrupt the long central helix and allow for the flexibility near the hinge [10]. In [193] it has been suggested that a number of side-chain interactions near the hinge region in calmodulin can affect the stability of the hinge. Interestingly enough, in this study they also mention that the hydrogen bonding interaction between Asp78 and Glu82 likely affects the hinge motion of calmodulin and perhaps it contributes to the relatively unbent form in the open (calcium bound) calmodulin state ¹³.

¹³As a side comment, we found that in the closed (calcium free) state which has more of a bend in the central helix and partial unfolding near the hinge [10, 47, 106, 193], this hydrogen bond is not present, whose absence (and absence of additional other N-H to C-O helical hydrogen bonds near

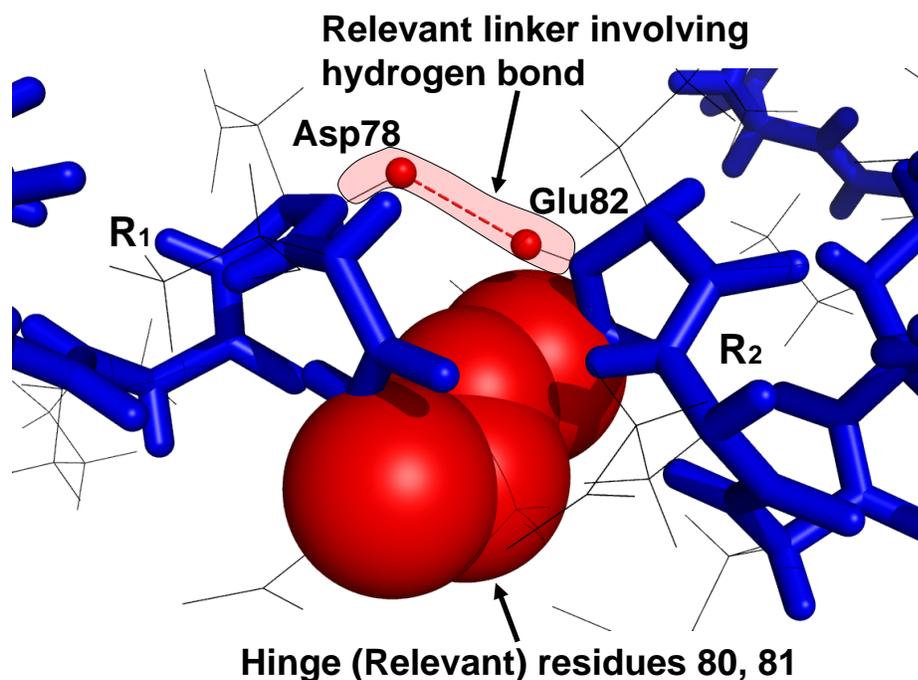


Figure 4.7: In calmodulin (pdb id: 3cln), in addition to the backbone hinge in residues 80 and 81, we also find that there is a relevant region between the two rigid clusters R_1 and R_2 that is formed by a hydrogen-bond between the backbone N-H of Glu 82 and C-O of Asp 78, and is responsible for the very tight and highly constrained hinge motion in calmodulin, with only 1 DOF. The identification of this hydrogen bond as a relevant linker that is constraining the motions between the two hinge-bending clusters is also suggested in [10]. It is important to mention that hydrogen bonds near the hinge can be expected to break and reform as the hinge motion occurs, although there exist protein examples where (preserved) side-chain hydrogen bonding interactions form part of the hinge between two domains [75].

The calmodulin example illustrates the type of detailed predictions we are capable of making using our Hinge-prediction algorithm. ■

Having the ability to predict both main-chain and side-chain interactions that affect hinge motions, and to an atomic and bond detail as illustrated with calmodulin, can add important insight that are generally not detectable by non-experimental computational methods. Furthermore, prediction of the actual DOF in the hinge (the hinge) is likely responsible for the more bent form of the central helix, where the hinge in the closed state also had more DOF.

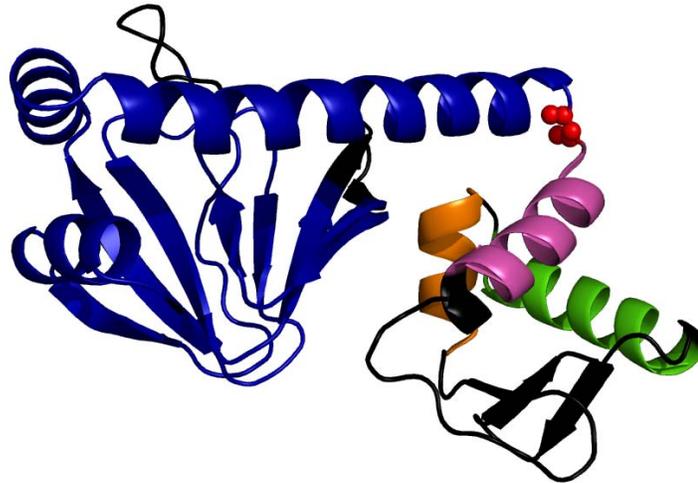
using only a single structure is not provided by any other techniques to our best knowledge, and this makes our method very unique and useful [53]. Since our goal is not to study any specific protein or how their function is affected by the hinge motions, we refocus our attention and proceed with the discussion of other features of hinge motions we can detect using our hinge prediction method.

As with Inorganic pyrophosphatase and calmodulin (see Table 4.1 for more examples), these proteins have the most typical and perhaps the simplest types of hinges, as the hinge is located along a single region along the backbone.

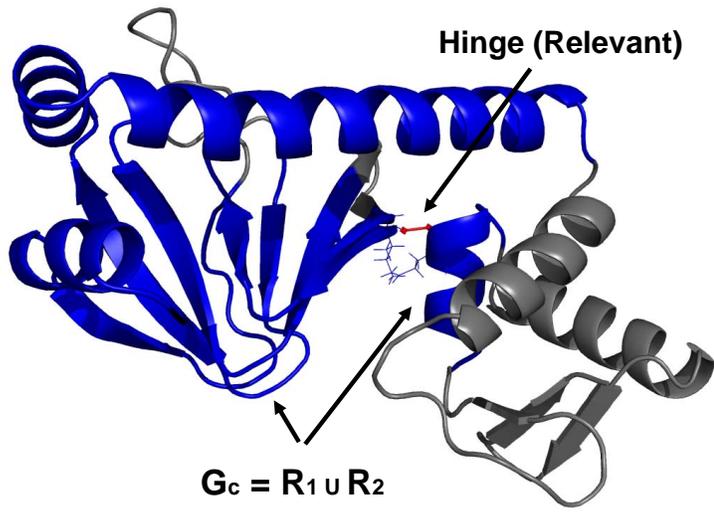
Catabolite gene activator protein

The protein catabolite gene activator is known to undergo hinge-bending inter-domain motions [75]. This protein is not found in HAG but makes an interesting case to test our algorithm on. What makes this protein very interesting in terms of its motions is that the hinge-bending motions occur primarily through noncovalent interactions between segments which are remote along the backbone chain. This is rather rare for a hinge-bending protein, as a majority of the identified inter-domain hinges in proteins are located in the backbone of the protein, with some possible additional stabilizing hydrogen bonds near the hinges, as we have seen with calmodulin. We suspect that the reason this protein (and many others) are excluded from HAG as there is no agreed hinge location in the backbone and hinge predictors as we mentioned earlier are designed to primarily predict only backbone hinges.

In Figure 4.8 (a) we have shown the rigid cluster decomposition from FIRST on catabolite gene activator (pdb id: 1g6n). One region which has been suggested in literature where some bending occurs is around residues 137 between the blue long helix and the purple helix. The Hinge-prediction algorithm was able to predict that



(a)



(b)

Figure 4.8: Hinge-prediction algorithm applied on a catabolite gene activator protein that has a hinge formed by non-covalent interaction remote along the backbone. Rigid cluster decomposition (pdb id: 1G6N) (a). There is a relevant region between blue and purple clusters, which is a mobile region, indicated with red spheres, residue 137 (b). The hinge that is reported in the literature is formed by non-covalent interactions between the blue and the orange clusters [75]. When we apply the Hinge-prediction algorithm, defining the large blue cluster and the orange clusters as the core, we find that there is a relevant linker between residue 60 and 174, through a hydrogen bond (b). We find this hinge to have 3 DOF. This hydrogen bond is well preserved and particularly important for the hinge bending motion of this protein [197].

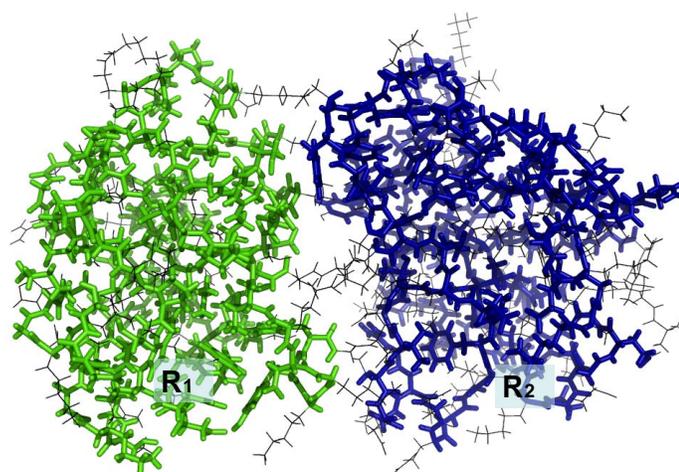
large blue cluster and the purple cluster are in a coordinated motion, with backbone 137 found to be a relevant linker, which is shown in red spheres.

However, more importantly it has been found that this protein primarily undergoes a large hinge bending motion somewhere between the large blue cluster and the orange helix, so it was proposed that there must be another hinge site [75]. Since there are no local backbone direct connections between the large blue cluster and the orange helix, non-covalent interactions may be constraining the two relative motions of two clusters and acting as a hinge. What we found when we ran the Hinge-prediction algorithm, is that there is a relevant linker composed of hydrogen bonding interaction between residues 60 Ile and 174 Gln (Figure 4.8 (b)). This hinge is found to have 3 DOF. Our finding is confirmed by authors of [75, 197], who have computationally and experimentally found that hydrogen bonds between these remote regions along the protein sequence, and in particular this specific preserved hydrogen bond that we found to form a relevant linker is particularly important for the hinge bending motions of this protein.

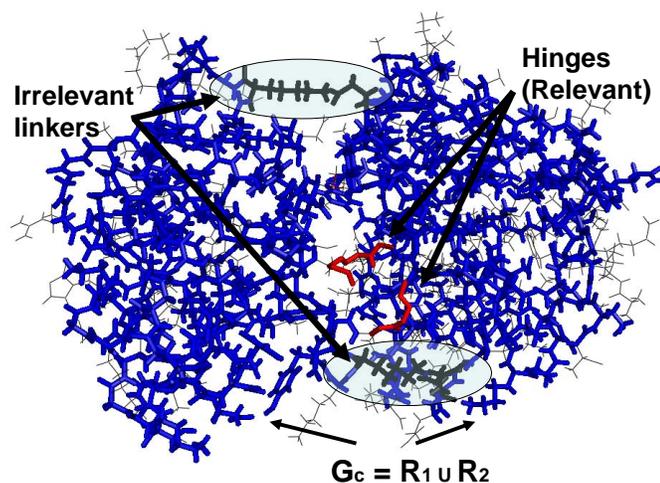
This example illustrates that our methods can successfully capture hinges that are not only caused by certain flexible (relevant) main-chain dihedrals but also hinges that are created by preserved noncovalent interactions through formation of hydrogen bonds between the two rigid clusters.

LAO Binding protein

In Figure 4.9 we have shown the rigid cluster decomposition (a) and our hinge prediction on lysine/arginine/ornithine (LAO) binding protein (pdb id: 2lao) which is a hinge with 1 DOF. This protein is different from the previous protein as it has multiple linkers connecting the two rigid clusters. The Hinge-prediction algorithm gives a very good match of the HAG hinges, for both hinge linkers. Our predictions for

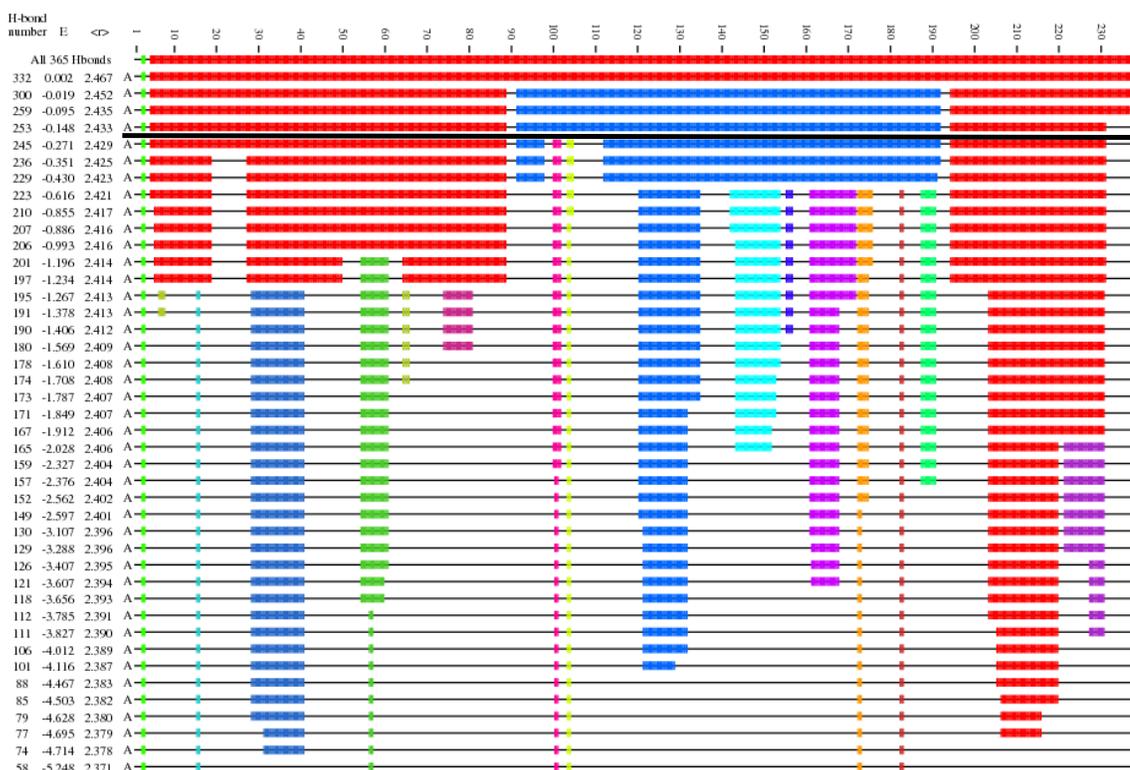


(a)



(b)

Figure 4.9: Hinge prediction on LAO binding protein (pdb id: 2lao). FIRST finds that most of the protein is composed of two large rigid clusters R_1 and R_2 (a), with 4 linkers between the two clusters. Hinge-prediction algorithm finds the two correct hinge locations (b) which agree very well with the HAG hinges, which are coloured in red in the stick representation (residues 89-91 and 192-194). The other two linkers (encircled on top and bottom) are irrelevant and do not constrain (restrict) the motions of the two combined rigid clusters. This hinge has 1 DOF. This example illustrates the advantage of utilizing the Relevant regions detection algorithm into hinge predictions, as only relevant linkers should be declared as hinges and lead to coordinated motions of the two cluster motions, while the irrelevant linkers have no impact on the hinge motions.



(a)

Figure 4.10: Hydrogen bond dilution plot (pdb id: 2lao). This protein breaks down into two rigid clusters (red and blue blocks) and the two intervening hinge linkers (residues), which are predicted to be relevant, can be roughly visually inspected. The dilution plot is a useful tool for selecting the appropriate energy cutoff prior to FIRST analysis. For instance, a cutoff of -0.25 kcal/mol (shown as a thick black line) can be used to correctly detect the hinge location. Note how the hinge is retained over a wide value of cutoffs.

the hinges are residues 89-91 for one linker and residues 192-194 for the other linker, whereas HAG reported hinges are residues 90-91 and 192-193.

In Figure 4.10 we have presented the hydrogen bond dilution plot for LAO binding protein, and how we select the hydrogen bond energy cutoff to obtain the rigid cluster decomposition prior to applying the Hinge-prediction algorithm. In this protein, the protein breaks into two rigid clusters early in the dilution, and once the two clusters are formed any choice of energy cutoffs up to -0.616 kcal/mol can be

used to predict the hinges, indicating that the hinge is stable over a range of cutoffs. See Chapter 1 and Chapter 6 for more discussion on dilution plots and the energy cutoff values.

One important observation about this protein is that there are two other linkers (through non-covalent interactions) between the two clusters. These two linkers have somewhat longer length and the Hinge-prediction algorithm correctly discards them both and declares them as irrelevant linkers. This example perfectly supports our intuition and early speculation in this chapter, that some linkers can just abstractly connect the two clusters without restricting their motions. Without the careful adaptation of the Relevant regions detection algorithm into the Hinge-prediction algorithm, it would have been very arduous to apply the counting arguments we gave above with perfect bridges, and capture which linkers are relevant and form the hinge.

Adenylate Kinase

Adenylate kinase is a hinge-bending protein that has two independent hinge locations which occur on two segments of the backbone. In Figure 4.11 we have provided the rigid cluster decomposition and our hinge predictions (pdb id: 2ak3). This protein is unique in comparison to the previous test cases, as it has two hinges which involves more than two rigid clusters. Normally we recommend picking the two largest nearby rigid clusters and testing them for the possible hinge location. In this case the largest rigid cluster R_1 is in close proximity and connected through a backbone with another large rigid cluster R_2 consisting of a long helix, however the largest rigid cluster R_1 is also connected by a flexible linker to another rigid cluster R_3 ¹⁴. As both R_2 and R_3 are spatially very close to R_1 , we chose to test out both combinations ($G_c = R_1$ and R_2) and ($G_c = R_1$ and R_3) for possible hinges. To test for hinges we ran the

¹⁴It has been reported that the two helices corresponding to the rigid clusters R_2 and R_3 close like two neighbouring fingers with respect to the large rigid domain corresponding to rigid cluster R_1 [60]

Hinge-prediction algorithm two times, once for each pair. In both cases we find that there is a hinge, and our predictions of hinges span residues 124-125 (Figure 4.11 (b)) and 158-159 (Figure 4.11 (c)) which are again very closely matched by the hinge location in HAG, which are residues 125-126 and 161-162. Each hinge has 3 DOF, that is the relative DOF between R_1 and R_2 is 3, and the relative DOF between R_1 and R_3 is also 3.

As a side comment, in Figure 4.11 (d) we ran the Hinge-prediction algorithm simultaneously by defining the core to be all three rigid clusters. As expected, both hinges are still correctly identified. By incorporating all three rigid clusters into the core, this may speed up the predictions as it requires only one run of the Hinge-prediction algorithm. However, the DOF count would not be as easily extracted as there are now more than two rigid clusters, but in this example it could be still done (since there is no coordinated motion between R_2 and R_3) by drawing 6 pebbles to R_1 and the remaining free pebbles to the other two rigid clusters (see Figure 4.12 (a)).

Even though R_2 and R_3 are each in a coordinated motion with R_1 through two identified hinges, they are not in a coordinated motion with each other. We can get six free pebbles on R_2 and six free pebbles on R_3 simultaneously. Alternatively, from visual inspection we can see that the only possible relevant linker between R_2 and R_3 will go through R_1 (this was also checked). But, this could not be a relevant linker, as the linker between R_1 and R_2 forms a hinge of 3 DOF and the linker between R_1 and R_3 also forms a hinge of 3 DOF. Both of the hinges, in terms of counts, can be viewed as a perfect bridge of length 3, and since there are two such bridges, then R_2 and R_3 are connected by a bridge of length of 6 which is not short enough to force R_2 and R_3 into a coordinated motion (see Figure 4.12 (b)).

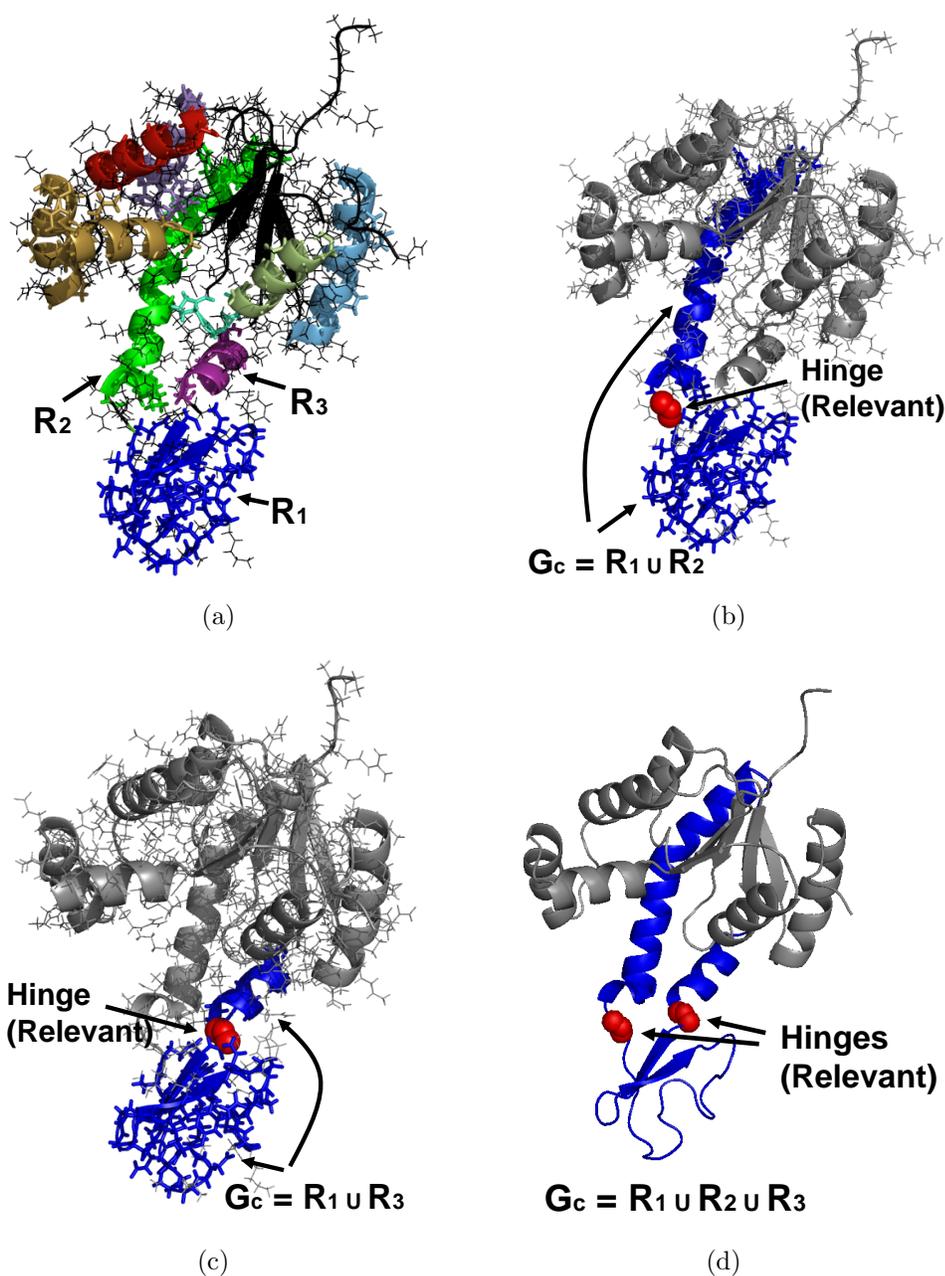


Figure 4.11: Hinge prediction on adenylate kinase (pdb id: 2ak3). FIRST finds numerous rigid clusters (a). The largest rigid cluster R_1 (blue) is in close proximity to two other clusters R_2 (green) and R_3 (purple). In the Hinge-prediction algorithm, we test both pairs (R_1 and R_2) (b) and (R_1 and R_3) for the hinges (c). Both runs find hinges (158-159 for R_1 and R_2 and 125-126 for R_1 and R_3) that closely match HAG hinges. Each hinge has 3 DOF. Alternatively, one run of the Relevant regions detection algorithm combining all three clusters into the core finds both hinges (d).

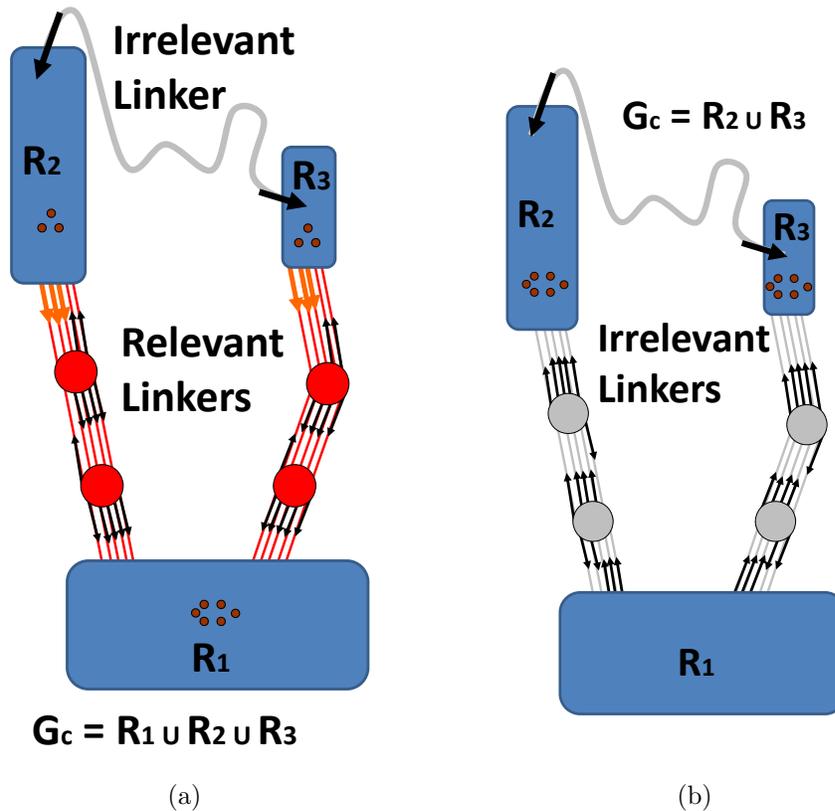


Figure 4.12: A simplified representation of the output of Hinge-prediction algorithm on adenylate kinase (pdb id: 2ak3), with pebble game output. Rigid cluster R_1 and R_2 are in a coordinated motion and have three relative DOF and are connected by a relevant linker forming the hinge. In the graph we have represented this linker with a perfect bridge of length 3 (a). R_1 and R_3 are also in a coordinated motion with 3 DOF, connected with a perfect bridge of length 3 (edges and vertices in red). If we declare core to be composed of R_2 and R_3 , each gets simultaneously 6 free pebbles and then there is no relevant linker connecting these rigid clusters, so they are not in a coordinated motion (b). The bridge between R_2 and R_3 passing through R_1 (which can be viewed as a single vertex) is a perfect bridge of length 6 which is not short enough to become relevant as per discussion above.

Even though we have already obtained a very good prediction of the location of the hinge, this protein has 5 other rigid clusters of significant size, so we posed a question: Is there a hinge motion between any other pair of rigid clusters? For completeness, we tested for hinges between all possible pairs of clusters. As anticipated,

we found that no two other rigid clusters are involved in a hinge motion (coordinated motion), that is their linking region was never relevant.

This protein illustrates the usefulness of performing the careful analysis with the pebble game algorithm, as it is able to successfully distinguish the relevant linkers from irrelevant linkers which was confirmed by correct prediction of hinge locations which is in accord with the experimentally determined locations as annotated in HAG.

Schematic Molecular model

To further demonstrate the power of using this approach to study coordinated motions of rigid clusters, in Figure 4.13 we have constructed a toy molecular graph model which contains three rigid clusters forming the core, and a collection of linkers connecting them, with the output of the pebble game algorithm also shown. It is not immediately obvious which linkers are relevant in this example. By taking the three rigid clusters and combining them into the core, it turns out that there is only one linker, between rigid clusters R_2 and R_3 that is a relevant linker and would be called a hinge, and is constraining their relative motions, as there are 5 relative DOF between R_2 and R_3 (i.e. freeze or ground say R_3 , then R_2 has only 5 DOF as is also evident that one pebble got pulled away from R_2). So, the motions of R_2 are coordinated with motions of R_3 . The other linkers between R_1 and R_2 , and between R_1 and R_3 are all irrelevant, so these clusters can freely move with respect to other (i.e. freeze R_2 and/or R_3 , then R_1 still has 6 full rigid body DOF).

We will discuss in more detail in the concluding section of this chapter the possible applications of this work to mutation studies, more specifically which mutations could alter the hinge motion and in general coordinated motions between the rigid clusters. In the spirit of the example in Figure 4.13 or in the protein examples such as the LAO binding protein, removing or adding some edges in the relevant

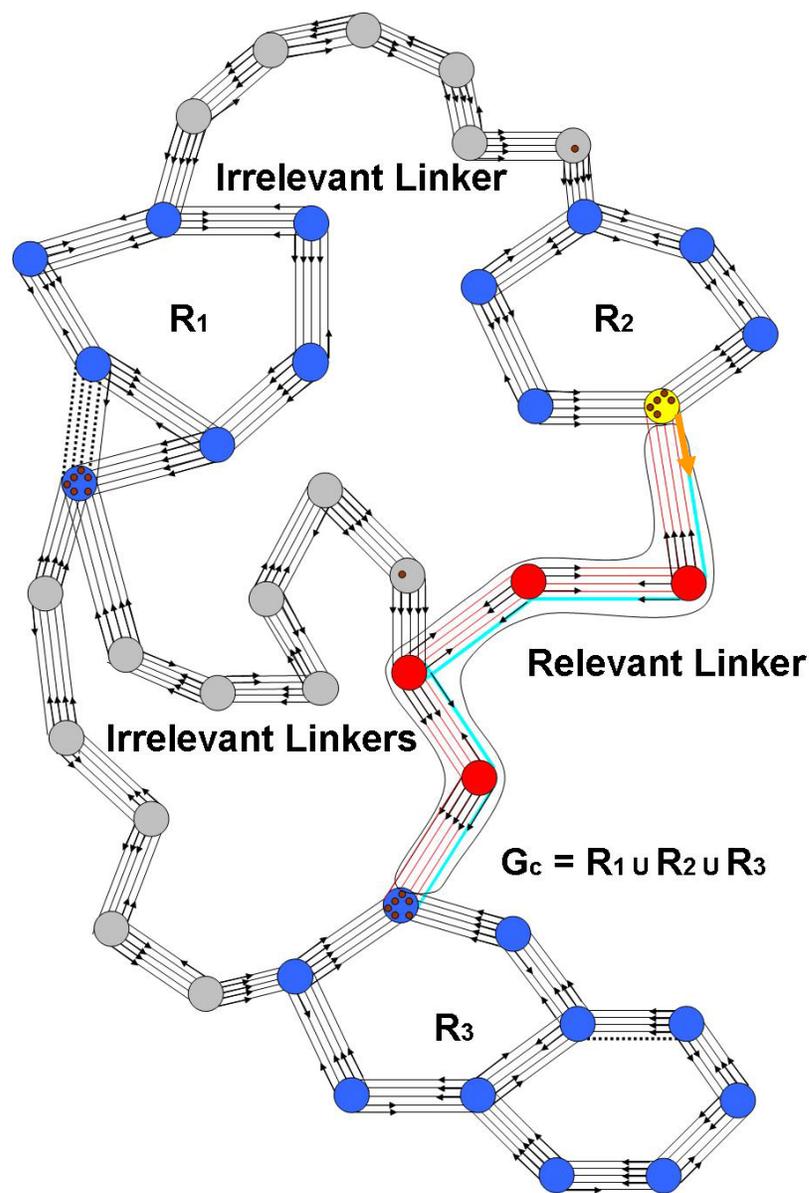


Figure 4.13: The three rigid clusters R_1 and R_2 and R_3 are connected with several perfect bridges of various size, with some intersecting bridges. The output of the pebble game with relevant regions detected is shown. The relevant region analysis indicates that only R_2 and R_3 are in a coordinated motion with one DOF with one relevant connecting linker (hinge) shown in red. The remaining linkers are irrelevant, which indicates that $(R_1$ and $R_2)$ and $(R_1$ and $R_3)$ are not involved in a coordinated (hinge) motion. Note that there are no directed paths between R_1 and R_2 and between R_1 and R_3 . The entire structure has 19 free pebbles, 17 which belong to the core.

linkers (hinges) would change the hinge motions and the number of DOF between rigid clusters R_2 and R_3 . The removal/insertion of edges (bonds) can be thought of as simple mutation in the relevant linker (i.e. residue insertion/deletion).

Immunoglobulin FAB Arm

As the last test case, we want to illustrate that the Hinge-prediction algorithm is capable of detecting existence of motions that are not pure hinge bending motions between the two clusters, but still involve constrained relative motions between the two clusters, which in the literature may have other classifications (for instance shear motions or motions with unknown mechanisms [36, 61, 62]). We should keep in mind that we are not generally able to differentiate among the types of motions, as our tools (i.e. pebble game algorithms) rely on snap-shot (i.e. infinitesimal) predictions of rigidity and flexibility. To actually classify the type of motion, simulations would have to be performed (which include additional constraints such as collisions) with other methods such as FRODA or coarse graining MD-simulation methods (see more in concluding section).

We applied the Hinge-prediction algorithm on the FAB arm region of the immunoglobulin, which is not classified as a typical hinge motion but is known to undergo relative domain motions [117]. Immunoglobulins (or antibodies) are proteins produced by the immune system in response to foreign molecules, and are central part of the immune system [15]. In Figure 4.14 (a) we have shown the rigid cluster decomposition from FIRST on the immunoglobulin protein (pdb id: 1igt). We have highlighted one of the two FAB arms (antigen-binding fragments)¹⁵ (for structural and functional details of immunoglobulins, and common terminologies, see [15] for

¹⁵It is common practice to view the immunoglobulin as a ‘Y’ shaped structure, with two arms, known as FAB arms, and the body (or trunk) in the middle holding the arms. The Fab arms play an important role in the function of this protein, as the tips of each arm (called variable regions) bind to antigen molecules [15].

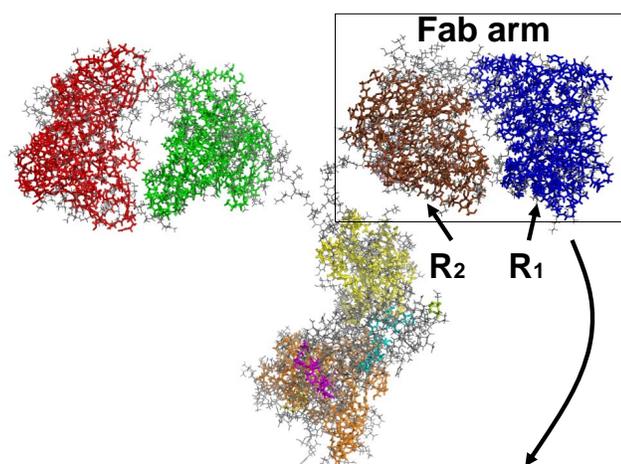
instance). The output of FIRST indicates that the Fab arm is composed of two large rigid clusters (one cluster corresponding to the so called ‘variable region’ domain and the other one to the ‘conserved region’ domain), which are connected by flexible linkers known as the ‘elbow’ region.

The Hinge-prediction algorithm finds that elbow region indeed constrains the motions of one cluster relative to the other cluster and can be declared as a hinge (see Figure 4.14 (b)). Both linkers are predicted to be relevant linkers joining the two clusters. We find that there are 5 relative DOF between the two clusters (i.e. 11 free pebbles on the core). The two domains in the FAB arm are known to be in a coordinated (restricted) motions due to the flexible connecting elbow region, and Lesk et al. [117] have suggested that the elbow region acts as a ball-and-socket joint between the two domains, although some (small) shear motions (translations) might be possible [81]. The ball-and-socket joint (shoulder and hip joints are examples of ball-and-socket joints) possesses three rotational DOF and any shearing motions would add additional DOF, which would mean that the elbow region in the FAB arm allows for a relatively high DOF motion (likely 4 or 5 DOF) between the two clusters. This means that the elbow region should have an increased flexibility, and this is supported by our finding that there is 5 DOF between the two clusters. Literature also suggests that these ball-and-socket motions (and any possible additional shearing motions) as in the elbow region are common domain-domain motions [117]. So, our methods could be applicable to a wider class of proteins and predict the location between the two clusters that is constraining their motions, that may not specifically be listed to have hinge motions, but can undergo these other types of motions. ■

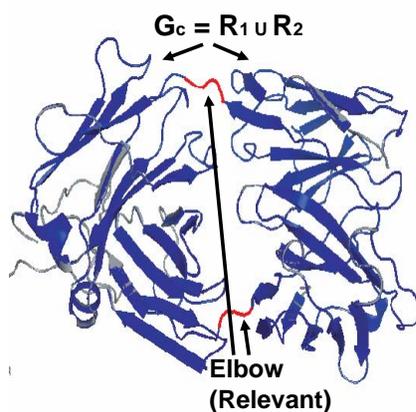
Remark. In certain situations (conditions) it is challenging to predict the hinge location. We give a short discussion of this.

Table 4.1: Hinges predicted on hinge-bending and other proteins using Hinge-prediction Algorithm 4.4.1, and compared to the HAG (when available).

Protein	PDB ID	HAG residues	Algorithm 4.4.1 hinge residues	DOF
Inorganic pyrophosphatase	1K23	189-190	189-190	3
Calmodulin	3CLN	80-81	80-81	1
Catabolite gene activator	1G6N	N/A see [75, 197]	hydrogen-bond linker between 60 and 174	3
LAO Binding Protein	2LAO	90-91, 192-193	89-91, 192-194	1
DNA Polymerase beta	2BPG	88-89	90-91	4
Ribose binding protein	1DRI	103-104, 235-236	235-236	2
Adenylate Kinase	2AK3	125-126, 161-162	124-125, 158-159	3, 3
Bence Jones Protein	3BJL	110-111	110-114	2
Immunoglobulin (FAB ARM)	1IGT	N/A see [117]	C107-108, D112-113	5



(a)



(b)

Figure 4.14: Application of Hinge-prediction algorithm on FAB arm of immunoglobulin (pdb id: 1igt), typically not classified as a hinge-bending protein, but the two domains in the FAB arm are known to undergo coordinated motions [117]. Rigid cluster decomposition on the immunoglobulin, with one of the two FAB arms shown, which are primarily composed of two rigid clusters (a). Hinge-prediction algorithm finds that there are two relevant linkers going through the backbone of two separate chains (residues C107-108 and D112-113 where C and D indicate chains C and D), which corresponds to the ‘elbow region’ in the FAB arms (b). There are 5 relative DOF between the two clusters.

Since we are using FIRST to identify the two (large) rigid clusters that will be processed through the hinge detection algorithm, our analysis will be dependent on the initial output from FIRST. For instance, FIRST might not identify the two

rigid clusters as the protein may be too flexible. This may occur due to the presence of irregular secondary structures and suboptimal hydrogen bonding geometry. It is also possible that the protein data bank file may be of a poor quality and contain numerous errors (which often occur [152]), so incorrect or not sufficient number (or too many) hydrogen bonds are identified by FIRST, leading to a poorer result of the rigid cluster decomposition. So the output of FIRST can be sensitive on the quality of the structure available. All of these factors could affect the FIRST rigidity/flexibility predictions and the initial selection of the two rigid clusters in the Hinge-prediction algorithm.

FIRST may also predict the protein to be composed primarily of a single large rigid cluster, so it would not be possible to identify the second rigid cluster, or the suspected hinge region may be part of a rigid cluster, for instance, in ribose binding protein, see Table 4.1. As our method only declares hinges that are constraining the motions of the two clusters, if there are too many DOF between the two rigid clusters (i.e. 6 DOF), we would not declare such connections (i.e. they are irrelevant) as hinges. Also, in cases where there are more than 2 large rigid clusters (for instance as was the case with adenylate kinase), it is not obvious which pair of clusters should be tested for a hinge, so more than one pair may need to be checked. ■

In Table 4.1 we have summarized the hinge predictions on the case proteins just discussed and several more proteins, with the corresponding HAG or literature hinge locations, and the relative DOF in the hinge. Note for adenylate kinase there is a 3, 3 relative DOF indicated, which means that between one pair of rigid clusters, there is 3 DOF, and between the other pair there is also 3 DOF. In Ribose binding protein one hinge was missed when compared to HAG, as FIRST has identified those residues to be part of one of the two rigid clusters. The other hinge in this protein was predicted exactly.

Overall, we find that the hinges we identify on these wide variety of hinge-bending proteins and the non-covalent forming hinges are in very good agreement with the HAG annotated hinges and other literature reported hinge locations. Looking over the Table 4.1 it is important to note that majority hinges (8 out of 11) that we have identified with our Hinge-prediction algorithm are composed of two residues. This is remarkably consistent with previous studies (as was mentioned above) which found that most frequently occurring hinges are composed of two residues [40, 51, 75].

4.6 Concluding remarks and future work

In this chapter we have presented a novel way of predicting hinges between rigid protein domains. The hinge-prediction algorithm 4.4.1 primarily uses the pebble game algorithm and the extensions of the relevant regions detection algorithm to predict hinge locations. Our algorithm has demonstrated very good predictions on a number of selected different protein structures with diverse types of hinges. The predictions are well matched with the HAG hinges (collection of visually annotated and experimentally determined hinges from the literature) and other literature based hinges on proteins not listed in HAG. In some tested proteins we obtained exact matches with HAG hinges. In addition, the hinge-prediction algorithm was able to detect coordinated motions between two clusters (domains) applied on a protein that is not listed in HAG but is known to undergo constrained relative domain-domain motions.

Our intent here was to demonstrate that careful extension of the relevant regions and the pebble game algorithm, together with theoretical considerations from the rigidity theory, is useful for detection of hinges. Our goal was not to undertake a large bioinformatics type of approach. However, due to the computational efficiency of the pebble game algorithm, the hinge-prediction algorithm we have introduced is

able to give rapid predictions of hinges requiring only a single structure (pdb), which makes it applicable for high-throughput analysis. So, in the future work the hinge-prediction algorithm could be used in a larger bioinformatics type of a study (see more comments below).

There are several key unique features and advantages of the Hinge-prediction algorithm. First of all, we are able to achieve sub-residue accuracy in hinge identification, which goes in further depth than the traditional residue-level hinge predictions, which only report backbone location of the hinge. That is, with this algorithm we are able to predict both backbone and any side-chain non-covalent interactions and identify the exact atoms and bonds that form the hinge, which are important (i.e. relevant regions) for coordinating (constraining) the motions of the two clusters. This was demonstrated on several proteins. Having the ability to detect hinges which can be formed by both covalent and non-covalent interactions, our algorithm could also be applicable to larger oligomeric proteins (i.e. have multiple chains) where the two rigid clusters can be found in separate chains of the protein which are often held together with many inter-domain non-covalent contacts [3].

We are able to predict which linkers are constraining the motions of the two clusters, and should correspond to the true hinge location, from other irrelevant linkers which may just loosely connect the two clusters but have no impact on their coordinated motions. If one were to start with the claim that any flexible linkers connecting the two clusters is a hinge location, our approach will reduce the number of any possible false positive predictions, associated with long irrelevant linkers. Although there are protein examples where a rigid helix can form part of the hinge region between the two rigid clusters [59] (which in terms on $6|V| - 6$ count acts as a single rigid body (atom)), it is hard to imagine too many other scenarios that hinges of very long lengths can form relevant linkers. Our observation on the hinge-bending

proteins that were tested finds that hinges most often have two residues, which is consistent with previous studies.

This is also the only method (to our best knowledge) that can obtain the number of degrees of freedom of the hinge motion (i.e. relative DOF between the pair of rigid clusters). Our algorithm is also capable of detecting hinge motions that may include hinge motions between more than one pair of clusters.

With this method we are also able to predict the existence of non-canonical two-rigid body coordinated motions in proteins that are reported in the literature (e.g shear, elbow motion in immunoglobulin). So, the hinge-prediction algorithm is applicable to proteins that have general domain-domain coordinated motions. In future work this may allow us to detect previously unknown coordinated domain-domain protein motions.

With the rapid ability to detect hinge motion prediction and the DOF count of the hinge, this novel Hinge-prediction algorithm can be a useful tool for further classification and understanding of the hinge motions in proteins. We can envision that this algorithm could also be used as an alternative and supplementary tool in classifying hinge motions within the Database of the Macromolecular Movements.

We can also foresee wider bioinformatics approaches using the developed Hinge-prediction algorithm. For instance, it might be useful to develop a database of predicted hinge motions or general domain motions built up using our algorithm. In addition to predicting hinges for a large class of proteins and labelling general irrelevant linkers that are not restricting (coordinating) the motions between two domains, it would also be possible to probe how often hinge motions are of a few DOF, say 1, 2 or 3 vs more DOF like 4 and 5. The database would annotate the predicted locations of hinges (relevant linkers) on various known and more importantly unknown protein domain motions, so we could start to mine for any general patterns. The

DOF count in the hinge motion may also provide new information about flexibility and how constrained are specific domain motions. This could potentially be used to test hypothesis about protein motions or to try and interpret the experimental results [212].

Having the ability to find only the relevant linkers and detect in-depth detailed sub-residue accuracy prediction of hinges, may have important consequences for anticipating the influence of mutations on the hinge motions and general protein rigidity and flexibility. So, with this fast relatively simple computational algorithm, we could potentially address an important biological question: Which mutations cause the change in the hinge motion and which mutations are not important and have no impact on the hinge motion or on rigid domain-domain motions in general? Instead of just randomly selecting amino acids to be mutated, we expect that mutations in the predicted hinges (relevant linkers) are likely to alter the hinge motions and possibly the functioning of the protein. The general ideas of the impact of mutations on protein motions can also have important implications in allosteric regulation of enzymes (see next Chapter). Mathieu et al. [124] have recently shown that it is possible to design allosteric sites in the hinges, where binding an allosteric ligand at hinge sites (that are removed from the active sites) can interfere with the flexibility of the hinge (i.e. make the hinge more or less flexible), which can alter the underlying hinge motion and subsequently modulate and alter the protein activity.

The natural next step in the future work would be to go beyond the statics and infinitesimal motions that are inherent in FIRST and the pebble game algorithm, and study the actual underlying motions. In this context, having the ability to predict the hinge locations to a significant detail as it was seen in this chapter, can be very useful when trying to simulate the motions of two two clusters (i.e. MD simulations, FRODA), where the focus of the simulations should just be on the two

clusters with the relevant connections (hinges) in between them while other irrelevant regions could be ignored. So, the rigid clusters obtained from FIRST and the Hinge-prediction algorithm developed in this chapter could serve as an initial coarse graining step in simulating hinge motions and other domain-domain motions, which would be useful both for the focus of the relevant pieces (hinges) and for speeding up the simulations. In the spirit of the general motivation of the Relevant regions, we are reducing the total DOF of the system to a significantly smaller number of essential DOF corresponding to the core and its relevant regions. In this way, the simulations could be performed in the reduced search space, so the computer resources could be focused only on simulating the relevant regions and the relevant motions of the protein that are responsible for the underlying hinge motions.

The Relevant region analysis and our extensions has proved to be very useful in predicting hinges in proteins. We now turn to another application of these general ideas to a very important biological phenomena of allostery in proteins.

Chapter 5

Rigidity model of Protein Allostery

5.1 Overview

In protein science, allostery is an important regulator of many cellular processes [68] and it has been suggested that all dynamic proteins are potentially allosteric [3, 73]. Allosteric control of protein function is a mechanism by which a binding event at one part of the protein has an effect on other parts of the protein, very similar to the behaviour of a telecommunications network, where receivers, transmitters, and transceivers can communicate with one another across significant distances [68]. The binding of a ligand at the allosteric site (i.e. a site other than an active site) triggers a conformational change that is somehow transmitted through the protein to cause a rearrangement and alteration in the shape at the active site [19, 190], which is often located a substantial distance away from the allosteric site. The exact mechanism of this functionally driven signalling through allosteric structural change propagation is elusive and not well understood.

Motivated by this important phenomena of shape transmission through a protein, in this chapter we will use concepts and methods from the combinatorial rigidity theory and develop three types of rigidity-based allosteric models. We will show how

(often very small) conformational and/or rigidity change at one site can propagate through the graph of sample toy model structures and affect the conformation and rigidity (i.e. shape), often dramatically, at a second distant site. We hypothesize this is how some proteins perform their allosteric transmissions.

Once definitions of the three kinds of rigidity-based allosteric communication are stated, we will develop the corresponding three combinatorial rigidity-based *allostery-detection algorithms*. These novel algorithms can detect transmission of rigidity between distant sites in the molecular graphs (i.e. proteins). We will first illustrate these concepts and algorithms (which use various pebble game extensions) using the select mathematical examples, some of which we have physically built as toy models (i.e. bar and joint models, Dreiding Atom tool kits – see Figures 5.2, 5.3 and 5.5), some are designed on the underlying graphs of molecular frameworks. We will provide various theoretical results, with detailed proofs about the rigidity-based allosteric communication models. We will mathematically verify a previous hypothesis from other researchers on features of allosteric communication pathways.

Once the theoretical concepts and algorithms are introduced on toy model templates and general graphs, we will move to biological applications of the algorithms. We will apply the rigidity-based allosteric models and algorithms on some actual protein structures from the Protein Data Bank, including the important class of receptor proteins called G-protein coupled receptors, which are identified as showing allosteric communication.

All of the work appearing in this chapter, definitions of rigidity-based communications, the corresponding algorithms, including the applications of the algorithms to proteins is new original work.

5.2 Allostery background

The activity of proteins and oligomeric proteins (i.e. proteins having multiple protein chains) often must be regulated so that they function at proper time and place [3]. The functional efficiency of most proteins depends on their ability to bind to a few specific molecules (ligands - from the Latin word *ligare* meaning “to bind”). One very important mechanism for regulating the activity of proteins (their binding), is by *allostery*, put forward originally by Monod et al. [19]. Allostery (sometimes called *allosteric effect*) can act as a general switch, very rapidly turning a protein from a functionally active state to a functionally inactive state, and vice versa [3].

Allostery involves communication between two distinct binding sites on the protein. The two sites somehow communicate so that the ligand binding at one site influences the binding of a ligand (which could be the same or different ligand) at a second site. The two sites can be located on the same protein chain or on different protein chains. It is generally opined that in allosteric proteins, conformational¹ and shape change at one site is cooperatively² coupled with a change in another site. Often one of the binding sites is the *active* site of the protein and the other site is called *regulatory* or *effector* site, which is sometimes simply called an *allosteric site* [3]³. The actual process of distant coupled conformational changes is largely not understood. Our goal is to propose rigidity base models for these couplings.

¹To erase any future confusion, the conformational change would exclude the trivial rigid-body motions of the entire protein (i.e. rotating and/or translating the whole protein).

²A commonly used term in allostery is *cooperativity*, where a protein exhibits *cooperative binding* if the affinity (i.e. ability to bind a ligand) of a binding site for its ligand is altered with the binding of the ligand at the other site(s). In nomenclature when the binding of the ligand at one site increases affinity for the ligand at another site, there is *positive cooperativity*, whereas in *negative cooperativity*, the affinity for the ligand at the other site is lowered [3, 15]. See below for the other commonly used terminology, such as *positive and negative allosteric regulation*.

³Active site in biology is part of an enzyme (protein) that usually forms a cleft or pocket on the protein where a ligand binds and undergoes a chemical reaction. A regulatory site on an allosteric protein is a site other than an active [3].

In more detail, small conformational and shape changes in the allosteric site⁴ that occur upon binding a ligand (which can be induced by breaking or forming bond(s)) can propagate through the protein, and somehow this information is transmitted to the active site, changing the conformation and shape of the active site, and in turn altering the affinity of the active site to bind ligands [73, 190]. A particularly important feature of allostery is that the binding sites are often ‘widely separated’ (topologically distinct), and in some cases the distance between them can span almost an entire protein [73]. The word allostery has Greek origins which nicely captures these two key features: *allos* meaning “other”, and *stereos* meaning “shape”, referring to involved conformational (shape) changes between distinct sites.

Allosteric control is biologically very important [3], and is an essential control mechanism for all organisms. The phenomenon of allostery is, for instance, prominently seen in transmembrane receptors, particularly in the G-protein coupled receptors, where the mechanical effect of ligand binding is mediated across the entire width of the plasma membrane [63] (see section 5.6.2). The biological literature reports that inaccurate allosteric regulation can often lead to disease, and that malfunctioning allosteric proteins are involved in cancer [34, 190]. In fact, allosteric communication in proteins is the focus of research in designing new types of drugs, which function by allosteric mechanism [37, 68, 73]. In the traditional drug development, drugs (small molecules) are designed to directly target the active site, whereas the new allosteric drugs are meant to target the allosteric site instead, and through an allosteric transition alter the activity of the active site. Since allosteric sites are potential novel drug targets, finding such possible sites is an important problem. It has already been shown that this novel approach to drug development is showing many significant benefits [73].

⁴Allosteric communication can also be initiated by a mutation of amino acid(s) at the allosteric site [68].

Binding of a ligand at an allosteric site can also be used to introduce a new binding site on a protein, where binding of a ligand to this new binding site could lead to changes in active site geometry, which can provide a further indirect mechanism of allosteric control [68].

To get a better visual sense of allostery, in Figure 5.1 we have schematically depicted the two main kinds of allosteric interactions in proteins [3]. In the first type (left half of the Figure), we have *positive allosteric regulation* [3] (or *allosteric activation*). In this schematic the two binding sites A and B are depicted as separated by a large distance. We assume that site A is the allosteric site and site B is the active site of the protein ⁵. We start with the ligand-free (apo) structure (Figure 5.1 (a) i) of the protein. When the ligand molecule (effector) binds to the allosteric site A (Figure 5.1 (a) ii), a signal is sent to a distant site B , which undergoes a conformational (shape) change (Figure 5.1 (a) iii). The propagated signal sent from A that caused a conformation change at the binding pocket at B , makes B now more susceptible to bind its ligand (Figure 5.1 (a) iv). Hence, in the positive allosteric regulation, which enhances the protein activity [3], a binding event at one site causes an additional site to open up for binding. A standard example of this type of allostery is the oxygen binding to hemoglobin; in hemoglobin there are actually four such oxygen binding sites [3, 16]. The binding of oxygen at one site induces a conformational change at other active site(s), which in turn increases the binding affinity for oxygen at other site(s).

In the second kind of allosteric interaction (Figure 5.1 (b) i-iv), called the *negative regulation* or *allosteric inhibition*, we see an opposite effect. In the schematic

⁵This schematic of allostery serves a general illustrative purpose of the types of allosteric interactions that we can see in proteins. However, we should note that in many allosteric proteins, often several binding sites are involved in allosteric communication. This can occur with single polypeptide chain (monomers) proteins but it is particularly true for many allosteric oligomeric complexes (having more than one chain), which often form symmetric protein assemblies [3, 69].

we start with a ligand bound at the active site B . A binding of a ligand at site A causes a conformational change at B , which in turn leads to decreased affinity for its ligand at B . So the binding at A leads to the active site B - located elsewhere in the protein - to become incapacitated [3].

Ligands that bind at allosteric site (sites A in Figure 5.1) are referred to as *allosteric effectors*. The effectors that increase the protein activity are referred to as *allosteric activators* (ligand at site A in Figure 5.1 (a)), while those that decrease the protein's activity are called *allosteric inhibitors* (ligand at site A in Figure 5.1 (b)).

In the schematic of Figure 5.1 we have encircled an imagined sub-region of a protein that connects the two sites, and represents the part of the protein that is responsible in mediating allosteric communication from site A to site B . These regions of the protein, sometimes called *allosteric communication pathways*, can be thought of as information processing networks, sending a signal from one site to another distant site. It is reasonable to anticipate that any mutations either directly in sites A or B or in the allosteric pathways can impair the communication between the two sites and consequently impact the protein function. Alternatively, it is conceivable that mutations in the residues that are not participating in the allosteric pathway or allosteric site(s) may not have an effect on the allosteric signalling between the two sites. Finding such protein allosteric pathways between distinct binding sites is an active and important area of research in the field of protein allostery [34, 190]. Despite of this intense interest, it still remains a poorly understood process. We will shortly discuss how our rigidity-based allostery models and the corresponding algorithms can be used to predict the potential allosteric communication pathways.

There are many other details about allostery that are studied in biochemistry [3, 190]. These would be distracting here and are not needed for our purposes. Only when required, we will introduce further biological concepts.

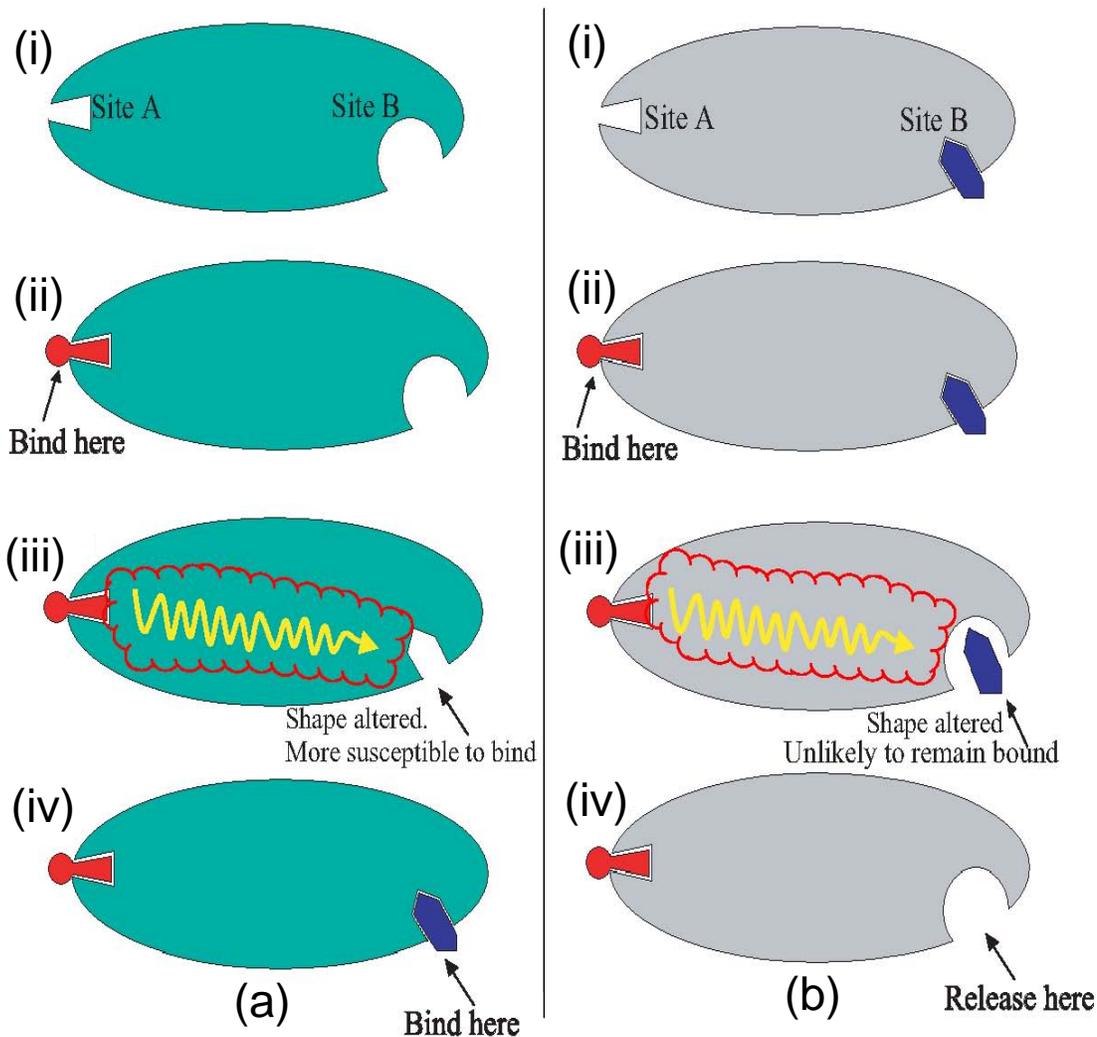


Figure 5.1: A schematic representation of the two types of allostery. On the left we have a *positive regulation*, where the binding of a ligand on allosteric site *A* (a(ii)), causes a conformational change on another distant (active) site *B* (a(iii)), so that site *B* is now more likely to recognize and bind its ligand (a(iv)). On the right side of the diagram we have another type of allosteric interaction called *negative regulation*. The active site *B* has a bound ligand. When ligand binds at site *A* (b(ii)) this causes the shape of site *B* to be modified (b(iii)), reducing the attraction for its ligand, so *B* is no longer ligated (b(iv)). The encircled zig-zag arrows indicate a sub-region of the protein, called *allosteric communication pathway*, that is responsible for transmitting the signal from site *A* to site *B*.

Even though it is becoming widely accepted that in allosteric proteins, conformational/shape change at one site is cooperatively coupled with a change in another

typically distant site, the allosteric mechanism propagating this structural change remains elusive and not fully understood [3, 68, 73]. Finding possible allosteric mechanisms should be very valuable in understanding and predicting how signals are regulated and transmitted across proteins. One of the difficulties is that allosteric-transmitting regions may involve multiple moving components and complex motions (and often some of these motions (unlike say hinge motions) may be very small), so even when the structure of an allosteric protein is solved in both active and inactive state structures, it is difficult to detect the mechanism or motions involved in shape transmission just by comparing the two structures.

We formulate some natural questions, which are often asked in allosteric studies, whose answers may help better understand the possible allosteric mechanism in proteins:

- (1.) How can a slight change in shape on one site propagate through the protein and change the shape on a distant site on another side of the protein?
- (2.) Given two (binding) sites A and B , will shape change at A cause a shape change at B ? i.e. are A and B allosterically coupled?
- (3.) Given a collection of 10 or more, for example, suspected allosteric sites, which of these sites is allosterically linked to the active site, so that binding at an allosteric site impacts the binding at the active site?
- (4.) Given two known sites that are involved in allosteric communication, which regions in the protein (i.e. which residues) between these sites are critical for this transmission to occur (i.e. what are the allosteric communication pathways?), and which regions are not important?

We hypothesize that tackling these allostery related questions from the rigidity-theoretical point of view may provide some new insight and methods to probe this biologically important phenomena.

5.3 A Rigidity approach to allostery

“I am never content until I have constructed a mechanical model of the subject I am studying. If I succeed in making one, I understand; otherwise I do not“.

– Lord Kelvin

One of the best known allosteric models is the MWC model, which was put forth by Monod, Wyman, and Changeux [19]. As with most allostery descriptions, this model also postulates that in allosteric proteins binding sites are linked in such a way that alteration in shape and conformation in one site is necessarily conferred to change at another (distant) site (and often more sites) [190]. Furthermore, this model claims that an allosteric protein can shift between two conformational states, the T (tense) or R (relaxed) states, where one state is better at binding its appropriate ligand (i.e. has higher affinity for ligand). In the MWC model, a shift between the relaxed and tense states is achieved by ligand binding to a site (allosteric site) that is different from the active site.

The coupled conformational and shape changes between distinct sites and the vocabulary of tense versus relaxed states in the MWC model is by itself suggestive that linked changes in rigidity and flexibility among the distant sites in the protein may have an important role in allosteric regulation. Thus, it would be natural to ask if changes in rigidity/flexibility in one site could propagate through the network (protein) and change the rigidity/flexibility on the other site?

Given a general protein representation as it is modelled in FIRST, we can reformulate some of the protein allostery questions from the previous subsection in

terms of rigidity of (generic) molecular frameworks and their underlying multigraphs. More precise explanation and definitions will be given in the next two subsections.

Starting with two binding sites ⁶ A and B , we pose these questions:

- (1.) If there is a change in rigidity at say site A , is there a corresponding change in rigidity at site B ? In other words, we want to know how are the degrees of freedom and motions of one site linked to the degrees of freedom of another site.
- (2.) More specifically, we can ask does rigidifying one site rigidity or at least reduce the internal degrees of freedom at another site? Equivalently, what is the maximum number of (internal) DOF that can be removed at site B by binding (i.e. adding edges) at site A ?
- (3.) If we freeze the motions of an atom or any rigid cluster in the site A , will this reduce the possible motions (DOF) at site B (i.e. are the motions of the two sites coordinated)?
- (4.) If we make A more flexible (i.e. break bonds, remove edges) can this add more flexibility to B ?
- (5.) If two sites are allosterically linked in this rigidity context, which regions (sub-graphs) are important for this transmission (i.e. what are the rigidity-based allosteric communication pathways?), in a sense that changes (i.e. breaking/forming constraints) in the communication pathway can modify or stop the communication of two sites.
- (6.) Are there any important rigidity or graph theoretical properties that can characterize rigidity-based communication pathways?

⁶So far sites were vaguely defined, using the protein terminology of binding sites. Shortly, sites will be defined in graph theoretical terms.

These are some of the general questions that we can list, and as we shall see shortly, they are naturally suited for analysis with rigidity techniques. We aim to show that various new extensions of the pebble game algorithm, which will be introduced later in this section, and their add-ons to FIRST implementation may be particularly useful in answering these questions. In addition to the new pebble game extensions that we will present, variations of the Relevant Regions detection algorithm will again come in very handy. The questions that were just posed will be stated more precisely and illustrated in more detail in the next section, and in addition some other specific problems pertaining to allosteric applications to actual proteins will be undertaken.

To introduce our rigidity approach to allostery, we begin by describing the ability of small mechanical toy models to transmit rigidity and flexibility across the structure between two sites.

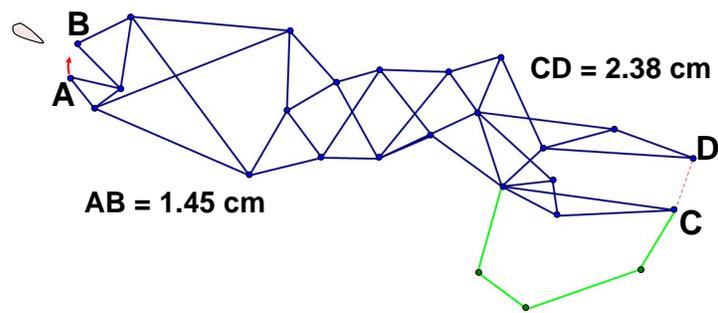
We start by looking at two 2-dimensional bar and joint frameworks, given in Figures 5.2 and 5.3). These frameworks have some of the desired allosteric properties and will serve a good illustrative purpose. The two frameworks are closely related but exemplify two distinct characteristics. Both frameworks were constructed using the Geometer's Sketchpad software. In addition to the computer model, we have also built an actual physical framework model. The physical model and the software model both exhibit similar motions and illustrate the same transmission of shape and DOF between sites.

The bar and joint framework F in Figure 5.2, if we exclude the green edges and vertices, has 1 non-trivial DOF. If we freeze or pin one edge (for instance an edge at B) to the ground in order to remove the 3 trivial DOF (a common mechanical engineering practice), this framework F is a 1-DOF linkage. Applying Laman's theorem (the $2|V| - 3$ count), we see that this framework F has 24 vertices and 44 independent

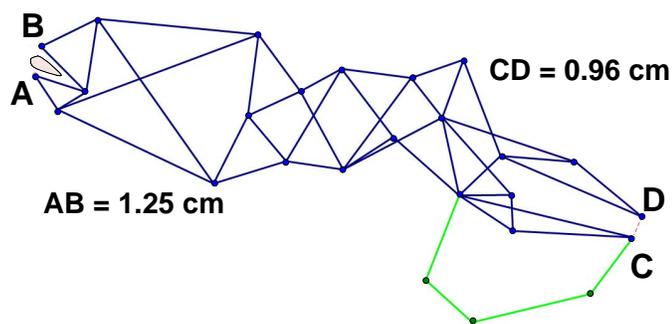
edges, which is one short from the minimum number of edges needed ($2(24) - 3 = 45$) to make it rigid.

In terms of allosteric behaviour, a particular feature of this framework, is that a slight motion at one end of the framework (A and B), will cause a dramatic change in conformation (shape) at the other end (C and D). If we slightly move vertex A towards vertex B, this will immediately cause vertices C and D to move very close to each other, which is evident both visually and by comparing the distances between A and B and between C and D. We can think of this framework as an example that resembles the positive allosteric regulation. A small ligand that fits at the binding pocket between A and B slightly pulls A and B together, which in turn leads C and D to also come closer together, making it more likely to dock a ligand in its pocket. In this example the closing motion at A and B has produced a closing motion at C and D. Note that the green edges have no impact on the motions of this 1-DOF linkage; the motions at A and B will cause the same motion at C and D with or without the green edges. So, we could say that the allosteric pathway between A and B only involves the blue edges and not green edges. Furthermore, if we rigidify the end A and B (add an edge between A and B), immediately the distance between C and D would become fixed. In this example the whole framework (excluding green edges) would become a large rigid cluster.

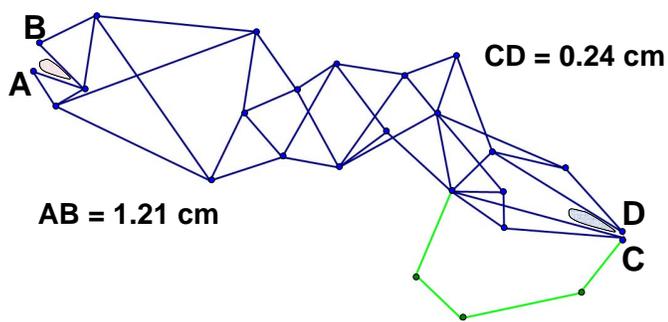
In the second example (Figure 5.3), we also have a 1-DOF framework. However, in this case a closing motion at A and B leads to an opening motion at the other end between C and D. This example resembles a hypothetical negative allosteric regulation, as ligand binding in a pocket between A and B (which slightly pulls A and B together) causes C and D to move apart, and as a result ligand is released at that end. Just like the previous example, rigidifying one end by adding an edge between A and B will cause the other end (C - D) to become rigid.



(a)



(b)



(c)

Figure 5.2: Transmission of shape and rigidity across a bar and joint framework illustrating allosteric behaviour. This framework (excluding green edges) has 1 non-trivial (internal) DOF. A slight (closing) motion between A and B is transmitted across a framework to cause an (opening) motion between C and D. Rigidifying A and B (i.e. place a bar between A and B) causes rigidification of C and D (i.e. stops the motion). The small imagined ligand binding at A and B, can promote ligand binding at C and D, resembling a positive allosteric regulation - allosteric activation. The green edges have no impact on this change of shape between the two sites, and in protein vocabulary would not be part of the allosteric pathway. Created with Geometer's sketchpad.

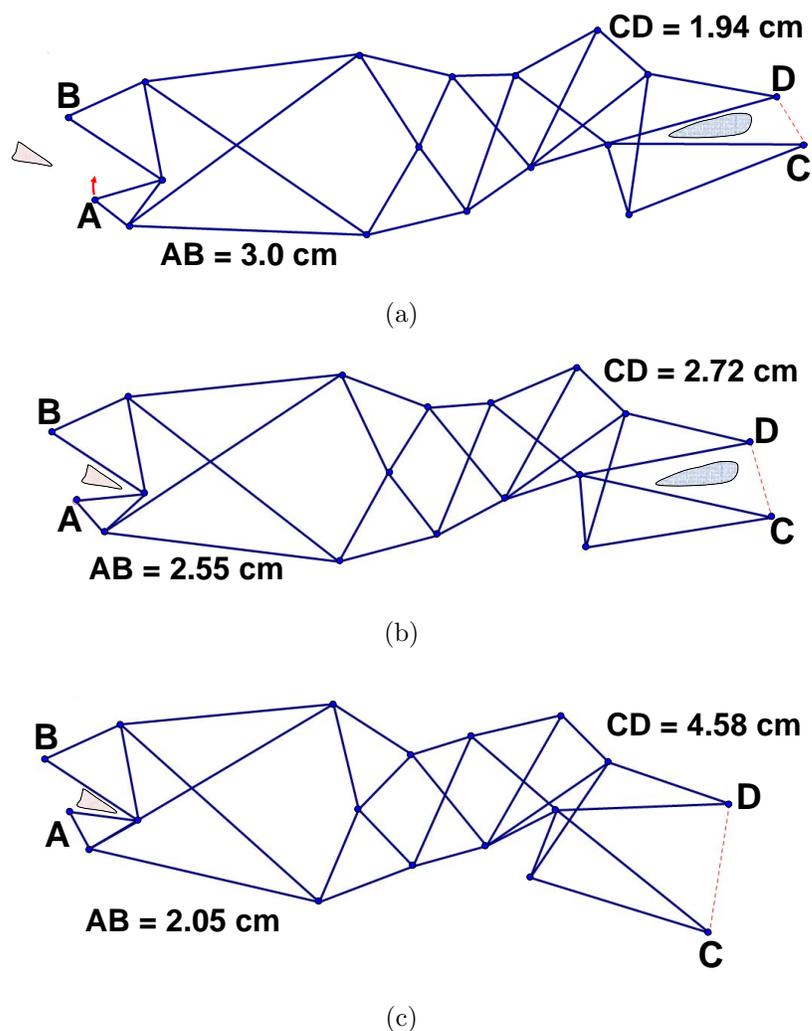


Figure 5.3: Transmission of shape and rigidity across a bar and joint framework illustrating allosteric behaviour. This framework has 1 non-trivial (internal) DOF. A slight (opening) motion between A and B is transmitted across a framework to cause a (closing) motion between C and D. Rigidifying A and B (i.e. placing a bar between A and B) causes rigidification of C and D (i.e. stops the motion). The small imagined ligand binding at A and B, causes a release of ligand at C and D, resembling a negative allosteric regulation - allosteric inhibition. Created with Geometer's sketchpad.

These two frameworks exemplify how slight changes in shape in a framework at one end can cause a (large) motion and change in shape at a distant site in the framework. In both cases rigidifying one site (simulating ligand binding) causes a reduction in the available DOF at the other (distant) site. Equivalently, squeezing

a pair of not connected vertices at one site (i.e. mechanically change the shape as a binding of ligand might) caused a shape of change at the other site. In Figure 5.4 we have shown a framework that does not have the type of rigidity-based allosteric behaviour that is seen in the previous two frameworks.

Even though the two example in Figures 5.2 and 5.2 are models of 2-dimensional bar and joint structures ⁷, they exemplify the general type of behaviour (i.e. change of shape and rigidification at a distance) that we expect is involved in allosteric proteins. In order to further highlight the idea of transmission of rigidity/flexibility (DOF) at a distance we turn to a model that is more directly related to molecules and proteins.

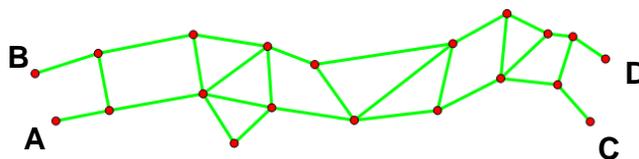


Figure 5.4: An example of a framework that does not have rigidity-based allosteric behaviour. In this framework any change in shape and rigidity at one site (A , B) does not impact the rigidity and shape at the other site (C , D).

Using the precise Dreiding Atomic Modelling kits (Figure 5.5), we have built and investigated several molecular toy models which also visually illustrate transmission of shape and rigidity/flexibility (DOF) between two sites. In these models, small motions, say bond rotations, at one end of the molecule can be visually seen to induce bond dihedral-angle changes at another end of the molecule, and rigidification at one site decreases the available DOF at the other site, exemplifying the shape changes between distant sites.

As an illustration, in Figure 5.5 we have displayed one Dreiding molecular structure model that we have constructed, that has some of these underlying allosteric

⁷Recall that molecules can still be represented using a special setting of 3-dimensional bar and joint frameworks with underlying G^2 graphs [207].

properties. This structure is composed of 22 atoms and 25 (rotatable) bonds (see corresponding multigraph representation), which by overall $6|V| - 6$ count ($|V| = 22$, $|E| = 5(25) = 125$) is one edge too few to be rigid (i.e. it needs $6(22) - 6 = 126$ edges). The $|E| \leq 6|V| - 6$ subgraph counts are satisfied on all subgraphs (i.e. it has no redundancy) so it has 1 internal DOF. A rotation of a bond at site A , causes a bond-rotation at site B , inducing bond dihedral-angle changes. Moreover, rigidification, for instance of site A (i.e. make that bond non-rotatable) causes site B to also become rigid. Rigidification of B would also rigidify A . This model exemplifies how rigidification of one site causes a transmission of reduced DOF at another distant site.

Note that in this model, rigidification of either A or B causes this whole structure to become rigid. Below we will look at some examples, where rigidifying one site causes a reduction in the number of DOF, but not complete rigidification of the other site.

This example is special as each atom (vertex) in this example forms its own rigid cluster, with the exception of a six-membered ring found close to site A . In this example there was no need to have large rigid clusters between the two sites for transmission of rigidity to occur (more precise definitions are given below). Having large rigid clusters between the two sites would essentially simplify or in some sense shrink the allosteric communication pathway as all vertices in a rigid cluster move as a single unified rigid body. As a preview and motivation to our discussion on proteins (below), we hypothesize that having large rigid clusters is one likely reason that allosteric effect in proteins can span such large distances, for instance there could be several rigid α -helices in the allosteric communication pathway (see discussion on calmodulin or GPCR's below). We speculate that using large rigid clusters to transmit

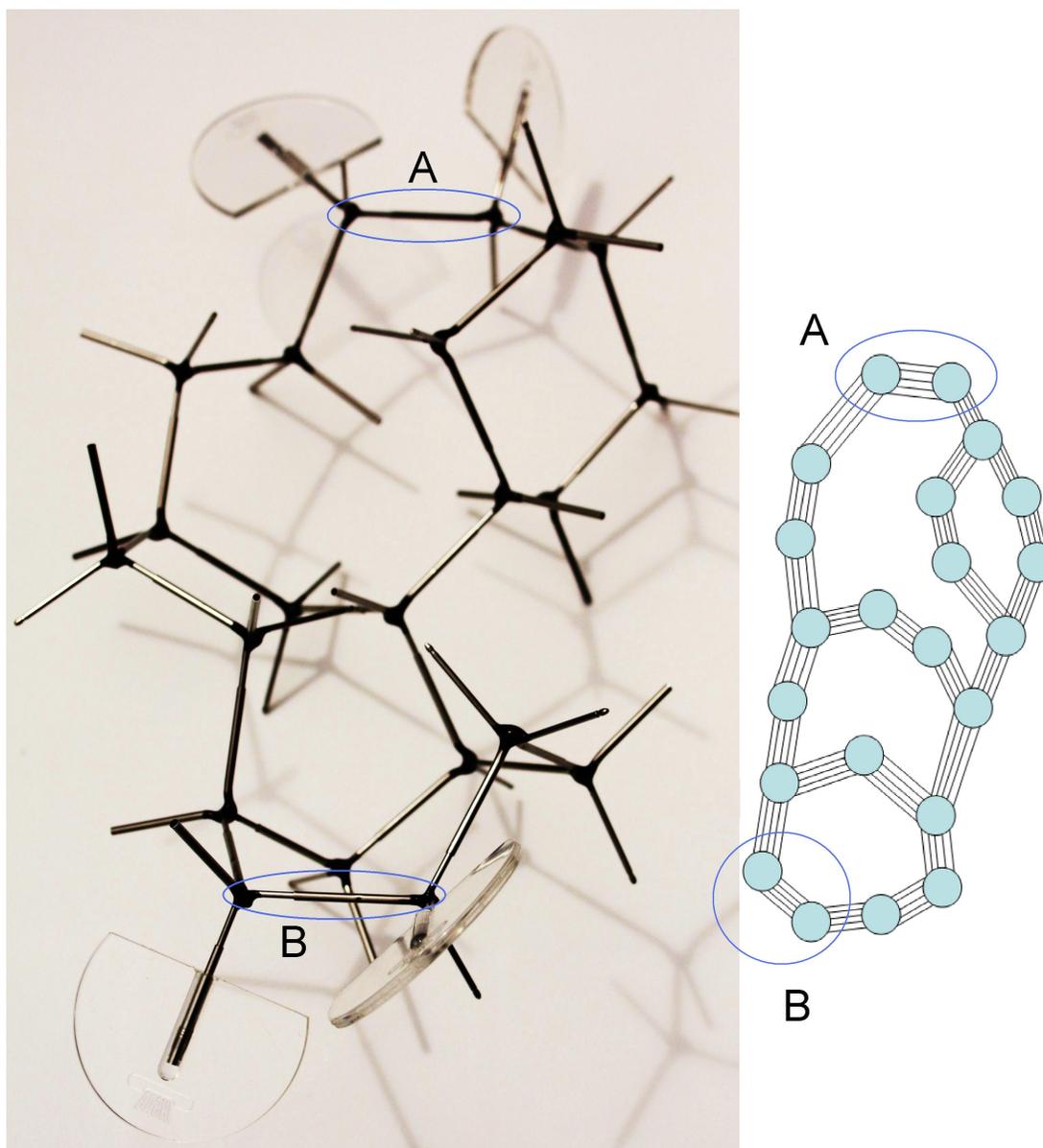


Figure 5.5: A Dreiding atomic model of a structure resembling allosteric behaviour. A motion (bond-rotation) at site *A* causes a propagation of change in shape to transmit across a structure, resulting in a motion (bond rotation) at site *B*. The effect works the other way (i.e. starting with a motion at *B*) as well. Rigidifying one site (decreasing DOF) causes a rigidification (decrease of DOF) of the other site. Moreover, if one atom in either site is frozen, the other site has reduced DOF, so there exists a coordinated motion between the two sites. This structure has 1 (internal) DOF. Underlying multigraph (molecular graph) is displayed on the right.

information across the structures can also be beneficial as most of the constraint network in the rigid clusters (i.e. hydrogen bonds) would presumably be preserved.

We state a few more properties using the physical model in Figure 5.5, some of which we will incorporate below in the rigidity-based allostery models. For instance, if we freeze the motions of one atom (i.e. freeze its 6 trivial rigid body motions) in one of the sites, there will be a corresponding reduction in the number of available DOF at the other site. More specifically, freezing one atom (rigid cluster) in either site causes a reduction in DOF at the other site from 7 DOF (this includes the 6 trivial DOF) to just 1 DOF. So there is 1 relative DOF between the two distant atoms, and from the previous chapter we could say that the two atoms (rigid clusters) are in a coordinated motion. As a further possible rigidity-based allosteric property, we note if there is an increase in flexibility in site B (for instance break a bond in B), then there will be a corresponding increase in flexibility (i.e. an additional DOF) that appears at the other end of the model ⁸. In the next section we will show how our algorithms can be used to detect these rigidity changes between distant sites.

In summary, both bar and joint 2-dimensional models and the molecular physical models, nicely illustrate the rigidity and shape transmission between two sites. Mathematicians in rigidity theory have been interested in allostery over the past several years and construction of small mathematical allostery examples in special bar and joint structures called ‘block and hole’ frameworks has been discussed in [46].

We now turn to more precise and detailed discussion and definitions of rigidity-based allostery, which will lead us to algorithmic solutions and subsequent applications to proteins.

⁸The bond in site A will still have 1 internal DOF corresponding to a rotation around that bond, but if we include another one neighbouring bond into A , breaking of a bond in B introduces an additional DOF that is propagated to the other side.

5.4 Rigidity models of allostery

Building on the templates from the toy models of the previous section which illustrated mechanical change of shape across the structures, we now sharpen our focus and introduce three rigidity-based allostery models. Once we have defined these models and concepts, we will introduce the corresponding algorithms (see section 5.5) that can be used to predict rigidity-based allosteric communication on multigraphs (and proteins). Two out of the three rigidity-based allostery models can be analyzed with modifications and new extensions of the Relevant regions detection algorithm, and one model will require additional new pebble game extensions.

The models and the corresponding algorithms that are discussed in this chapter could also be adapted to 2-dimensional bar and joint structures, and general body-bar/body-hinge structures. However, since we are mainly interested in protein allostery, our focus will be strictly on the molecular structures. Furthermore, as we are always working with generic rigidity, we continue to discuss rigidity/flexibility and rigidity-based allosteric behaviour of the underlying multigraphs. The examples that illustrate the three types of allosteric models will be given in section 5.5.2.

We hypothesize that the three rigidity-based allosteric models that we are about to introduce could represent possible ways that distant sites in a protein may communicate with each other and contribute to allosteric regulation. The three models are in some ways related to each other, but nevertheless have significant differences. So, restating the problem of rigidity-allostery and the allosteric models in several ways is important as it should enrich our understanding of how different types of rigidity induced communication (i.e. change of shape, rigidity/flexibility, coordinated motions) is propagated in the constraint graph. This should lead to a clearer understanding in how rigidity models of allostery can be used to explain allostery in actual proteins.

In all three models we start with a (molecular) multigraph G and two sites $A \subseteq G$ and $B \subseteq G$. We assume that A and B are disjoint induced subgraphs in G . As it is naturally expected, we are not interested in the overall rigidity/flexibility of G . We are only concerned with rigidity/flexibility of specified sites A and B . Until we specifically discuss proteins, we can continue to imagine that sites A and B symbolize two binding sites on a protein (i.e. allosteric site and active site).

In the first rigidity model of allostery, which we call *coordinated allosteric motions*, or *Rigidity-Allostery type 1 communication* or in short *allostery type 1*, we assume that the sites A and B are both rigid, but not part of the same rigid region. In terms of proteins, the rigid sites A and B can be as small as a single atom, a ring of five or six, or a larger rigid region. In other words, the relative DOF of A are only the 6 trivial rigid body motions (i.e. maximum free pebbles that can be reversed to A in the pebble game output on A is 6), and likewise for B . In allostery type 1 we are interested if the sites A and B are in a coordinated motion. This problem is closely related to the hinge predictions from previous chapter, but differs significantly in many ways (see below). In this model, we seek to answer the following question:

Rigidity-Allostery type 1 (coordinated allosteric motions): If we freeze one rigid site (i.e. remove or fix its trivial DOF), for instance site A , are the motions (DOF) restricted (reduced) at a rigid site B ?

If the possible motions (DOF) are restricted at B , we say that A and B are involved in allostery type 1 communication.

In other words, we are asking does freezing the six trivial DOF at A remove any DOF at site B . If no DOF at B are removed, then freezing A has no impact on the motions at B . We could clearly not have removed all 6 DOF at B , as that would mean that A and B were in a same rigid cluster. So the number of DOF that can be removed from B is between 1 and 5. Note that it does not matter which site

we freeze, the answer is the same. That is A and B are involved in allosteric type 1, is same as saying that B and A are involved in allosteric type 1 (see algorithm 5.5.1 for more details). When two sites are involved in allosteric type 1 communication (as with hinge motions), this number 1 to 5 represents the number of relative DOF between them. So, in allosteric type 1 the motions of one site is coordinated with the motions of the other site (i.e. A and B are in a coordinated motion).

For instance, in the physical model in Figure 5.5 if we define a site to be one atom (vertex) in site A and another site to be one atom (vertex) in site B , then these two sites (vertices) are involved in allosteric type 1 communication, and there is a one relative DOF between them (also see Figure 5.6 (c)).

As we had mentioned earlier, this is closely related to the hinge motions between two adjacent (large) rigid clusters, as technically hinge motions satisfy allosteric type 1 definition, however, there are several significant key differences. In allosteric type 1, the rigid clusters should be substantially smaller (i.e. a region near a binding pocket is typically small), whereas in hinges the rigid clusters can be composed of an entire domain often containing several secondary structures. Also, unlike in hinge motions, in allosteric interactions the two sites are typically separated by a great distance (averaging about 20 to 40 angstroms, and often reaching distances larger than 100 angstroms [68]).

The following analogy may further assist in understanding this type of allosteric communication. Imagine two people (sites) involved in a game of tug-of-war and assume that the (flexible) rope is long enough to significantly separate the two people. Tug-of-war is a game where two people or usually two teams pull on two ends of a long rope trying to move one another over a center line. If the rope is taut enough (i.e. not too flexible), then an initial hard pull (tug) by one person will cause a pull (motion) of the other person. So we could say that two (widely separated) people are

involved in a (allosteric) coordinated motion. On the other hand, if the rope is very loose (too flexible) and just dangling along the ground, then the initial pull at one end of the rope has no impact on the motions at other end, and we could say that two people are not involved in a coordinated motion. So tugging on one end of the taut rope propagates changes to the other end of the rope, and is one possible line of communication over a distance. The rope functions as an allosteric pathway between the two people (sites).

The tug-of-war analogy also illustrates that in allostery type 1, the two sites can communicate through a single pathway. In terms of the proteins we can think of the rope representing a single backbone chain that is connecting the two sites (in this analogy imagine ligand binding is causing an initial pull at one end). As the tug-of-war analogy suggests high cooperativity between distant sites, it is conceivable that sites in some proteins (see calmodulin below) could be involved in this type of allosteric communication.

Notice if in allostery type 1 we had allowed the two sites to be part of the same rigid cluster, then the two sites would in some sense be trivially in a coordinated motion. When one site moves the whole rigid cluster moves, including the other site. As we are not after this oversimplified communication between the two sites, which has been considered previously [146], we have assumed that the two sites are separate rigid regions. Allostery type 1 communication could be one possible mechanism how two sites (i.e. allosteric, active) communicate with each other, but additional forms of rigidity-based communications need to be considered.

In the second model of allosteric rigidity communication, which we call *Transmission of DOF or reduction of flexibility at a distance*, or in short *allostery type 2*, we assume that the two sites A and B are both flexible. In other words, the number

of internal relative DOF at A is positive⁹ and likewise for B . Thus, in the output of the pebble game the maximum free pebbles we can draw to A is greater than 6, and likewise for B . We denote the number of (internal) relative DOF at A and B by DOF_A and DOF_B , respectively.

We are not interested in the overall flexibility of the graph (protein), but only of the specified sites A and B . Roughly speaking, in allosteric type 2 communication we want to know if a reduction in DOF_A , has an effect (reduction) on DOF_B , and the other way around. The reduction in DOF at first site will be introduced by adding additional (independent) edges up to complete rigidification of that site (simulating a ligand). Note that in the schematic in Figure 5.1 (a) and (b) going from (ii) to (iv) represents allosteric type 2. That is, the initial ligand binding causes a reduction of DOF at site A , and the corresponding change in shape and reduction of DOF at site B .

More specifically, in allosteric type 2 we ask:

Rigidity-Allosteric Type 2 (Transmission of DOF or reduction of flexibility at a distance): If we rigidify one site, for instance site A , does this reduce the (internal) relative DOF of site B ?

To rigidify a site, say A , means we need to add enough (independent) edges among the vertices of A , such that DOF_A becomes zero. In terms of the pebble game (see algorithm 5.5.3), this could be achieved by first recovering the maximum number of free pebbles to A and placing the six free pebbles on any vertex in A and use the other remaining free pebbles to add additional (independent) edges in A (one edge per extra free pebble). If in rigidifying site A , DOF_B is decreased, then we say that sites A and B are involved in allosteric type 2 communication.

⁹Recall that internal (i.e. non-trivial) relative DOF of any vertex induced subgraph G_c is the maximum pebbles that can be reversed to G_c take away six, or equivalently maximum independent edges that need to be added to G_c to make it rigid. Also recall that the word ‘relative’ was used because it is DOF of G_c with the rest of the graph present.

We are interested in how many DOF can be removed at B by rigidification at A . This number we will denote as $\text{DOF}_{\overrightarrow{AB}}$. To adopt the vocabulary of allosteric communication in proteins, we will refer to $\text{DOF}_{\overrightarrow{AB}}$ as the maximum DOF that can *transmit* a reduction of DOF at B , upon rigidification of A . For brevity, we will say that the $\text{DOF}_{\overrightarrow{AB}}$ is the maximum DOF that can be transmitted from A to B . So, when two sites are involved in allostery type 2, there is a transmission of DOF from one site to the other ¹⁰. For instance, in the models from the previous section (Figures 5.2, 5.3 and 5.5) the two sites are involved in allostery type 2 communication. Rigidification of one site on one side, say A , had decreased the DOF at the other site B (specifically $\text{DOF}_{\overrightarrow{AB}} = 1$). In the next section, we will look at some examples of allostery type 2, where rigidification of say A causes a reduction of DOF_B , but does not lead to complete rigidification at B . See Theorem 5.5.5 which shows that $\text{DOF}_{\overrightarrow{AB}} = \text{DOF}_{\overrightarrow{BA}}$, in other words it does not matter which site is rigidified first, the same number of DOF are transmitted. So, we can equivalently say that A and B are involved in allostery type 2 communication or B and A are involved in allostery type 2 communication. We will discuss this in more detail.

In the third rigidity-based allostery model, which we call *Increase of flexibility at a distance* or *allostery type 3 communication*, we would like to know if an increase in DOF (i.e. additional flexibility) at one site will cause an increase in DOF at the other site. The initial increase in flexibility in the first site can be introduced by breaking a bond, or in terms of the underlying multigraph, as removing some edge(s). More specifically, we ask:

¹⁰Some confusion can arise in the reference to ‘transmission of DOF’. Rigidifying one site can only decrease DOF at another site, so perhaps a better word would be a ‘reduction’ of DOF at a distance. However, we chose to adapt the word transmission, from a commonly used expression of ‘allosteric transmission’ or ‘signal transmission’. Hence, there should be no confusion here or in the later section when we say in allostery type 2 $\text{DOF}_{\overrightarrow{AB}}$ degrees are transmitted from A to B .

Rigidity-allostery type 3 (Increase of flexibility at a distance): If we remove some edge(s) in the site A (increasing flexibility (DOF)), is there a corresponding increase in DOF at site B ?

If removal of edge(s) at the site A increases DOF_B , then we say there exists allostery type 3 communication. Note that unlike allostery type 1 and 2, in allostery type 3 it is important from which site the edges are removed. So, it is possible that removal of edges from A may increase DOF at B but not the other way around. To be consistent with the previous two types of rigidity-based communication, we will still sometimes say that A and B are ‘involved in’ type 3 communication, even though in type 3 communication we do not have the symmetric communication among the two sites as with the first two types. We will come back to the discussion of this shortly.

Note that in allostery type 3, we did not make any assumptions about rigidity/flexibility of sites A and B , whereas in type 1 both sites are rigid and in type 2 both sites are flexible.

5.5 Methods and Algorithms to detect rigidity-based allosteric communication

In this section we describe the algorithms that can be used to answer the questions posed under the three types of rigidity-based allostery models. We first deal with allostery type 1 and type 3 as these can be checked with extensions and refinements of the Relevant regions detection algorithm, and then we present an algorithm for detecting allostery type 2 communication. In the subsequent subsections, we will then apply the algorithms on a few examples, and verify several key properties of allostery type 2 algorithm.

5.5.1 Algorithms

The main underlying theme of the algorithms for all three types of allosteric communications is to provide an answer to these two important questions:

- (1.) Are sites A and B involved in rigidity allosteric communication?
- (2.) If sites A and B are involved in rigidity allosteric communication, what are the regions in the graph (i.e. allosteric pathways) that are important for this communication?

Recall that in allostery type 1 (coordinated motions), sites A and B are assumed to be rigid regions.

Algorithm 5.5.1 – Allostery type 1 (coordinated allosteric motions) detection algorithm:

Input: Molecular Multigraph $G = (V, E)$, induced disjoint subgraphs $A \subseteq G$ and $B \subseteq G$.

Output: Yes if A and B involved in allostery type 1 communication, no otherwise. If A and B are involved in allostery type 1, output the relative DOF between A and B , and region (allosteric communication pathway) over which the allostery type 1 communication occurs.

- (1.) Declare core $G_c = A \cup B$.
- (2.) Play $6|V| - 6$ pebble game on G .
- (3.) Find the enlarged relevant region G_R of G_c and get the maximum number of free pebbles on G_c .

If the maximum free pebbles in G_c is less than 12, then declare A and B are involved in allostery type 1 communication, no otherwise. Maximum pebbles on G_c

take away 6 is the relative DOF between A and B . Declare relevant region $G_R \setminus G_c$ (i.e. allosteric communication pathway) as the region over which A and B are involved in a coordinated motion, and all other (irrelevant) regions in G (i.e. $G \setminus G_R$) to not contribute to allosteric type 1 communication between A and B .

We now turn to a description of allosteric type 3 algorithm which basically utilizes the Relevant regions detection algorithm with slight modifications. In short, to find out if removing edges (i.e. break a bond) at site A increases DOF (pebbles) at B , we basically look for a relevant region from B and see if it contains some edge(s) in A . More specifically, if the enlarged relevant region G_R contains at least one edge in A , then allosteric type 3 occurs. If the relevant region of B does not include any edges in A , then removal of edges in A has no impact on the DOF at B , so allosteric type 3 is not possible.

As a side comment, it should be clear that if there exists allosteric type 3 communication (where A and B are not part of a same rigid cluster) then some selection of pair of rigid clusters or vertices (one from A , other from B) defining two new sites are involved in allosteric type 1 communication, as there must be a relevant region connecting these two sites. So, allosteric type 3 can in some sense be considered a subclass of allosteric type 1.

Note that allosteric type 3 specifically tries to answer if removal of edges in A can increase the number of DOF at B . If one is interested in the opposite effect, then that would have to be checked with the algorithm. In other words, it is possible that removal of edges in one site has impact on the DOF and motions on the other site, but not the other way around (see examples below). We will show some examples of this in section 5.5.2.

Algorithm 5.5.2 – Allosteric type 3 (Increase of flexibility at a distance) detection algorithm:

Input: Molecular Multigraph $G = (V, E)$, induced disjoint subgraphs $A \subseteq G$ and $B \subseteq G$.

Output: Yes, if removing edge(s) in A increases flexibility at B (i.e. A and B are involved in allosteric type 3), no otherwise. If the answer is yes, output the region (allosteric communication pathway) over which allosteric type 3 communication occurs.

(1.) Declare core $G_c = B$.

(2.) Play $6|V| - 6$ pebble game on G .

(3.) Find the enlarged relevant region G_R of G_c .

If G_R contains at least two adjacent vertices in A (i.e. vertices that are joined by an edge) then declare yes, removing some edge(s) in A increases DOF (pebbles) at B and A and B are involved in allosteric type 3 communication. Declare the relevant region of B (i.e. $G_R \setminus G_c$) as the region (allosteric communication pathway) over which allosteric type 3 communication occurs, and all other (irrelevant) regions do not contribute to this communication (i.e. can be ignored/removed and communication is unchanged).

We now state the algorithm for allosteric type 2 communication, which requires several different procedures and steps that were not needed for the other rigidity allosteric communications. Recall that in allosteric type 2, both sites A and B are assumed to be flexible. In order for A and B to be involved in allosteric type 2 communication, we need to check whether there is a transmission of a reduction of (internal) DOF from one site to another. Put another way, we want to know if rigidification of one site decreases the (internal) DOF at the other site.

In the previous two algorithms, a single maximum pebble draw and the subsequent (capped) failed searches out of the core (i.e. Relevant Regions detection

algorithm) were enough to check if the sites are involved in allosteric type 1 or 3 communication and to find the regions important for their communication. On the other hand, to check for allosteric type 2, a single pebble reversal (without special initial operations, see below) does not certify whether A and B are involved in allosteric type 2 communication. Once additional steps are checked to ensure that the two sites are using allosteric type 2 communication, then we continue to use the Relevant regions detection algorithm to find the regions (rigidity allosteric pathways) important for this rigidity-based allosteric communication.

Algorithm 5.5.3 – Allosteric type 2 (Transmission of DOF – reduction of flexibility at a distance) detection algorithm:

Input: Molecular Multigraph $G = (V, E)$, induced disjoint subgraphs $A \subseteq G$ and $B \subseteq G$.

Output: (a) Yes if A and B are involved in allosteric type 2 communication, no otherwise. (b) $DOF_{AB}^{\vec{}}$ - maximum number degrees of freedom that can be transmitted from A to B (i.e. number of DOF that can be removed at B by rigidifying A). (c) Region over which allosteric type 2 communication occurs (i.e. allosteric communication pathway).

- (1.) Play $6|V| - 6$ pebble game on G .
- (2.) Draw maximum number of free pebbles to vertices in A , and denote this number as A_{maxp} .
- (3.) Freeze all the free pebbles in A (i.e. they cannot be cascaded out of A).
- (4.) Draw maximum free pebbles to vertices in B . Denote the current number of free pebbles on B as B_{maxp}^A (i.e. pebbles on B while pebbles on A are frozen).
- (5.) Release frozen free pebbles in A .

(6.) Continue drawing maximum free pebbles to vertices in B , and denote this number as B_{maxp} .

(7.) Set $DOF_{\overrightarrow{AB}} = B_{maxp} - B_{maxp}^A - 6$.

(8.) If $DOF_{\overrightarrow{AB}} > 0$, find enlarged relevant region G_R of the core $G_c = A \cup B$.

If $DOF_{\overrightarrow{AB}} > 0$, declare A and B are involved in allosteric type 2 communication and there is transmission of $DOF_{\overrightarrow{AB}}$ degrees of freedom from A to B . That is, rigidifying A removes $DOF_{\overrightarrow{AB}}$ degrees of freedom at B . Declare the relevant region of the core $G_R \setminus G_c$ as the region (allosteric communication pathway) over which allosteric type 2 communication occurs. All other (irrelevant) regions $G \setminus G_R$ do not contribute to this communication, and could be ignored (i.e. removed) in studying the allosteric type 2 communication between A and B .

We elaborate a bit on the meaning of different counts that are obtained from the algorithm. For the sake of discussion, let us also imagine that one site, say site A , corresponds to the allosteric site, and site B an active site. We are rigidifying site A and tracking how many DOF are reduced at site B (the choice of which site is rigidified is not important, see Theorem 5.5.5).

First of all, A_{maxp} , which is the maximum number of free pebbles at A , measures the relative DOF of site A (prior to any binding event at A), and $A_{maxp} - 6$ gives us the internal relative DOF of A . In terms of proteins, this tells us how many DOF the binding site A has prior to a binding event at A . Similarly, B_{maxp} measures the DOF at site B prior to rigidification (or a binding event) at A . The value of $A_{maxp} + B_{maxp}^A$ is the maximum free pebbles on the core $G_c = A \cup B$, which tells us the relative DOF of A and B combined prior to rigidification of A . The value B_{maxp}^A is the number of free pebbles at B that cannot be cascaded back to A . When this number is positive, it indicates that rigidification of A is not able to cause a complete

rigidification of B . The $\text{DOF}_{AB}^{\rightarrow}$ count tells us if A and B are involved in allosteric type 2 communication and it gives us a quantitative sense of how many DOF are transmitted from A to B in rigidification of A . Another way to think of $\text{DOF}_{AB}^{\rightarrow}$ is a measure of the maximum possible degrees of freedom that can be removed at B by a binding event in A . Note that if the binding event at A does not completely rigidify A , it is possible that fewer than $\text{DOF}_{AB}^{\rightarrow}$ degrees of freedom are removed at B (see examples below).

In step 3 of the algorithm, freezing pebbles at A essentially mimics complete rigidification of A (the frozen pebbles, except six which must remain, can be removed and absorbed by adding one independent edge per pebble). Freezing pebbles at A is a more efficient way than actually adding maximum independent edges in A , as it is not immediately obvious without continuing to run pebble game algorithm which edges would need to be tested and added to A to make it rigid.

With these comments in place, an alternate algorithm, although perhaps not as efficient as the one we have given, could be described. Roughly speaking, one could add maximum possible independent edges in A so it becomes rigidified, call it A_{maxe} , add maximum possible edges at B while keeping new added edges at A , call it B_{maxe}^A , release edges that were added at A , and see how many more independent edges can be added to B , call it B_{maxe} . Then if $B_{maxe} - B_{maxe}^A$ is positive, transmission will occur, and this number would be the same as $\text{DOF}_{AB}^{\rightarrow}$.

In step 7, where we define $\text{DOF}_{AB}^{\rightarrow}$, we have removed 6 from $B_{maxp} - B_{maxp}^A$. The 6 represents the trivial rigid body DOF that are always present. More specifically, being able to cascade only up to and including 6 free pebbles from one site to another (step 6) is not enough for sites A and B to be involved in allosteric communication of type 2. On the other hand, an interesting observation is that if there is a cascade (transfer of pebbles - see Chapter 3 for definitions) of positive but less than or equal to

6 free pebbles from one site to the other, it is enough to tell us that there is allosteric type 1 between some pair of vertices or rigid regions in A and B . More specifically, even when $-5 \leq \text{DOF}_{AB}^{\vec{}} \leq 0$ ¹¹ some pair of rigid clusters or vertices, one in A and the other in B , are involved in allosteric communication of type 1 (coordinated motion) (i.e. there is a relevant region connecting them). When $\text{DOF}_{AB}^{\vec{}} > 0$, this would clearly still hold. So, allosteric type 2 can be viewed as a subclass of allosteric type 1. When A and B are involved in allosteric type 2 communication, some two rigid regions (or at least two vertices), one from A , the other from B , will be involved in allosteric type 1 communication.

One of the main goals of the Algorithm 5.5.3 was to detect if sites A and B are involved in allosteric type 2 communication. When $\text{DOF}_{AB}^{\vec{}} > 0$, then A and B are involved in allosteric type 2 communication. In order to extract the $\text{DOF}_{AB}^{\vec{}}$ count, we had to perform two carefully designed pebble draws to sites A and B (steps 2 - 6 of Algorithm 5.5.3). It is possible to slightly improve the efficiency of these steps by not considering the freezing operation in step 3 and 5. To do this requires testing the edges in A at the end of the pebble game run. Once all the edges in $G \setminus A$ are tested, prior to testing the edges in A we would draw the maximum pebbles to B , obtaining the B_{maxp} count, and then continue on testing edges in A . When the edges in A have been tested (and pebbles reversed to A), we would check how many pebbles were removed in B (i.e. the pebble count on B prior to testing A and after testing A), from which we could extract the count of $\text{DOF}_{AB}^{\vec{}}$. As in proteins, we might decide after the FIRST analysis what sets to choose as the sites A and B , so we did not emphasize any importance to this modestly improved efficiency.

¹¹If $\text{DOF}_{AB}^{\vec{}} = -6$ there is no relevant region connecting A and B , so no two pair of vertices (one from A and the other from B) are in a coordinated motion.

Remark. We can further relate Algorithm 5.5.3 to the Relevant regions detection algorithm, and an alternate procedure which can detect if allosteric type 2 communication is involved between A and B . Starting with two flexible sites A and B , we first rigidify site A and then define the core $G_c = B$. If the relevant region of the core contains at least one edge (adjacent vertices) in the new rigidified subgraph, then A and B are involved in allosteric type 2. We will not prove this, as it is not needed, but we can offer a rough sketch idea of the proof. If the relevant region of B contains some edge in A (after A was rigidified), then removing that edge should increase the DOF (pebbles) at B , equivalently when that edge was added to A , that had decreased the DOF at B . Notice we did not have to find $\text{DOF}_{\overrightarrow{AB}}$ count here.

However, it is clear that the $\text{DOF}_{\overrightarrow{AB}}$ count and other counts extracted from the Algorithm 5.5.3 are useful (see below). ■

We had commented a number of times that freezing of pebbles in step 3 of the algorithm mimics the rigidification of site A , which is a key initial step in the allosteric type 2 induced communication. This leads us to the important $\text{DOF}_{\overrightarrow{AB}}$ count, which can also give us a count of how many DOF are transmitted from A to B in rigidification of A . In respect to proteins, $\text{DOF}_{\overrightarrow{AB}}$ can tell us how the possible motions and the configuration space of the binding site B are restricted by a binding (rigidification) event at site A . More specifically, by comparing the counts B_{maxp} and $\text{DOF}_{\overrightarrow{AB}}$ we can get a quantitative sense of how much reduction in the DOF occurs at B when a (distant) site A is rigidified.

In addition to the final $\text{DOF}_{\overrightarrow{AB}}$ count, the individual counts A_{maxp} , B_{maxp}^A and B_{maxp} also contain useful information. For instance, A_{maxp} and B_{maxp} give us a sense of how flexible the binding sites (pockets) are prior to an allosteric effect (binding). If there are many DOF available at a binding pocket of the allosteric site, then a large number of constraints must be formed between the ligand and the binding pocket to

rigidify that site. In this case, we can speculate that perhaps a larger ligand (assuming it should form more hydrogen bonds and other interactions with the binding pocket), taking into consideration the shape and geometry of the binding pocket, would more likely send a rigidity-based allosteric signal to the other site. On the other hand, if the site has only a few DOF, a fewer constraints (edges added) are needed to rigidify the allosteric site, and depending on the shape and size of the binding pocket, a smaller ligand might be appropriate.

We now summarize two important properties that should be preserved under different runs of the Algorithm 5.5.3, which are verified in section 5.5.3. The key determining count $\text{DOF}_{\overrightarrow{AB}}$, a certificate if A and B are involved in allostery type 2 communication, should not be dependent on the initial pebble game output (i.e. the order the edges in G were tested in the pebble game - step 1) or on the order the draws were made (steps 2 - 6). More specifically, we will show that all the operations that got us to the count $\text{DOF}_{\overrightarrow{AB}}$, is not dependent on any initial pebble game operations. We also anticipate that it should not matter which site (A or B) is rigidified. In other words, the output of the algorithm should not depend on which site we draw and freeze pebbles on (steps 2 and 3). The transmission of a number of DOF (i.e. $\text{DOF}_{\overrightarrow{AB}}$ or $\text{DOF}_{\overrightarrow{BA}}$) between the two sites should be the same regardless of this choice.

Before we prove any of these observations and get deeper into several other important properties of this algorithm, we introduce several examples.

5.5.2 Examples

The following examples should help us better comprehend the definitions and communications involved in the three types of rigidity-based allostery models and they also nicely illustrate the outputs of the algorithms. The examples will be on relatively small multigraphs, so the outputs can, in most cases, be visually verified.

Only the first example explicitly focuses on allostery type 1, as in flavour it is very similar to the coordinated motions seen from previous chapter on hinge motions (for instance see Figure 4.13) between rigid clusters. As we said earlier, unlike hinges, in protein allostery the sites are typically very small and are often removed by a great distance. Furthermore, whenever allostery type 2 communication occurs there is some underlying pair of rigid sites (at least a pair of vertices) that are involved in allostery type 1, so those examples will be able to serve several purposes.

Example 1 – Figure 5.6

As a very simple example of allostery type 1 communication, let us consider the two sites A and B in Figure 5.6, where each site is represented by a single vertex (rigid body - atom). The output of the pebble game with pebbles reversed back to the two sites is also shown for easier demonstration (refer to Algorithm 5.5.1). In (a) the two sites are involved in allostery type 1 communication (coordinated motions), and there is 5 relative DOF between them (i.e. maximum pebbles reversed is $11 - 6 = 5$). So, if one site is frozen there is a reduction in the DOF (motions) at the other site. The two sites are in a coordinated motions, in other words one site is not freely moving (i.e. it is restricted) with respect to the other site. Sometimes we will say that the motions (DOF) of one site are coordinated with the motions (DOF) of the other site. The relevant region $G_R \setminus G_c$ of A and B combined (i.e. $G_c = A \cup B$) is coloured in red, and the irrelevant regions in gray. The communication between the two sites is only transmitted through the relevant region. Removal/addition of edge(s) in the relevant region causes stoppage/modification of the communication, respectively. More specifically, if any edge in the relevant region is removed then there would be no more relevant region connecting the two sites, as there would be more than 5 relative DOF between them, so A and B would no longer be involved

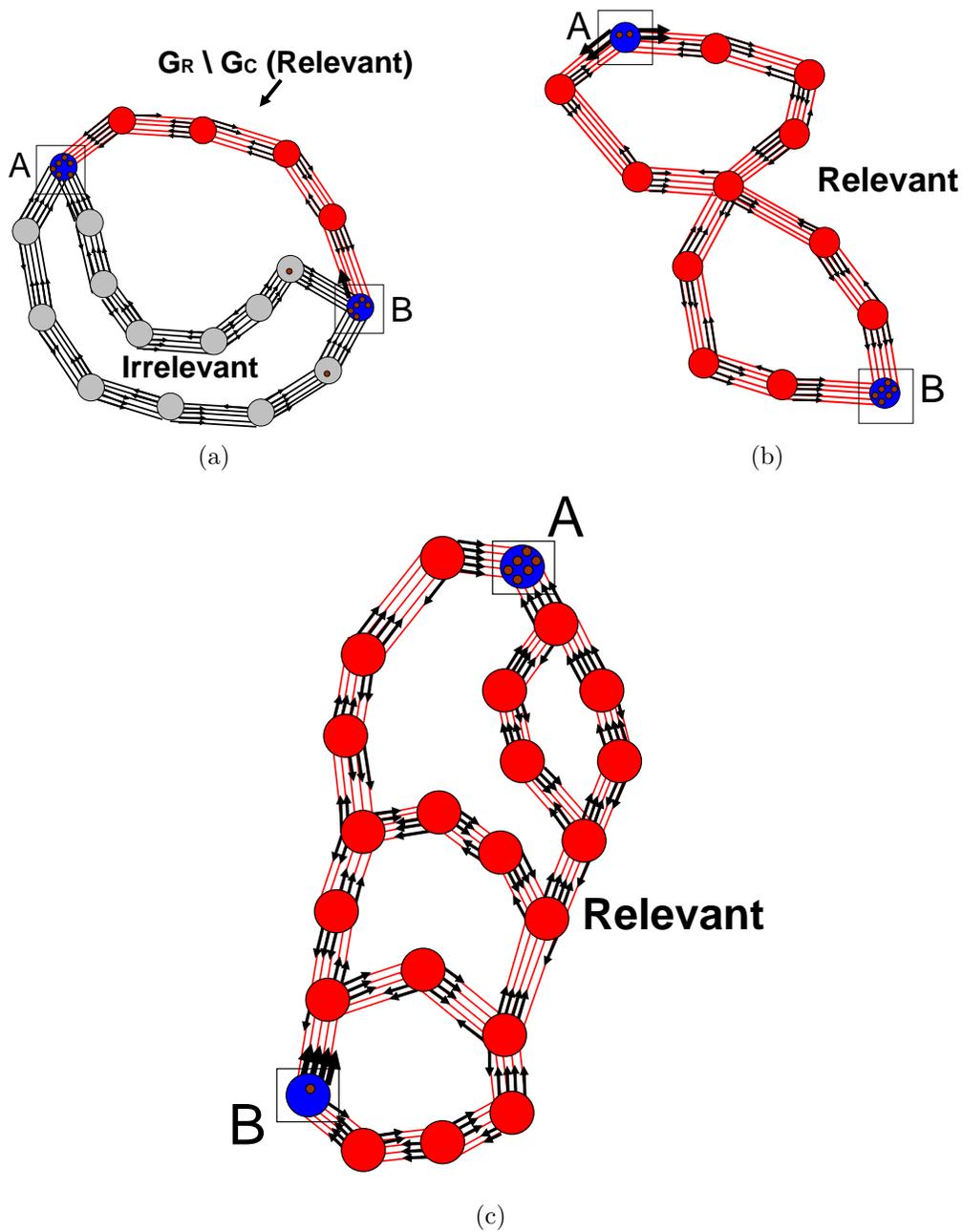


Figure 5.6: Pairs of sites A and B involved in allosteric type 1 communication (coordinated motions). Outputs of the pebble game with red vertices and edges corresponding to the relevant region $G_R \setminus G_c$ of the core $G_c = A \cup B$, and gray region is irrelevant and not important for the coordinated motions between the two sites. In (a), (b), (c) there are 5, 2 and 1 relative DOF between the sites. In (c) a pair of sites from the physical model from Figure 5.5 is selected.

in allosteric type 1 communication. The irrelevant region can be removed and the communication is still the same.

In the example in Figure 5.6 (b) the two sites are also involved in a coordinated motion and there is 2 DOF between them. Notice that it is possible to transmit the allosteric type 1 communication through a single cut-vertex (i.e. a vertex where the two rings of seven intersect - see below). We shall see later that in allosteric type 2 communication, transmission through a single cut-vertex (or rigid body) is not possible.

In Figure (c) 5.6 we have displayed the output of the pebble game on the multigraph of the physical model in Figure 5.5. Any pair of two vertices from original two sites are involved allosteric type 1 communication with one DOF between them. The whole graph is relevant with respect to these two vertices A and B and forms a transmission pathway (see output of Algorithm 5.5.1).

Example 2 – Figure 5.7

In Figure 5.7 we consider again the multigraph of the physical model from Figure 5.5. In this example we want to illustrate allosteric type 3 communication. The core is taken to be site A which is formed of 4 vertices and has one internal DOF (seven free pebbles). The relevant region of the core includes the site B , so removal of any edge in B would increase the internal DOF count at A by one (from 7 free pebbles to 8 free pebbles) (see Algorithm 5.5.2). Removal of two edges in B would increase the DOF count at A by two, and removal of more than two edges would not increase the DOF any further. It should also be evident that removal of any edge(s) anywhere in the relevant region of the core, would increase the DOF at site A . So if we are testing for allosteric type 3 communication, and only the active site is known, say site A in this example, then any region in the relevant region is a potential allosteric site.

This is a simple example of allosteric type 3 communication, where removal of edges (i.e. break a bond) at one site, which increases the flexibility of that site, causes an increase in flexibility at another site. Note if we define the core to be site B , then we would not find any relevant region, as removal of edges in A , and for that matter anywhere outside of B , does not add any additional DOF at B . So, in allosteric type 3 the communication is not symmetric as it is in allosteric type 1 and allosteric type 2, which we will shortly discuss.

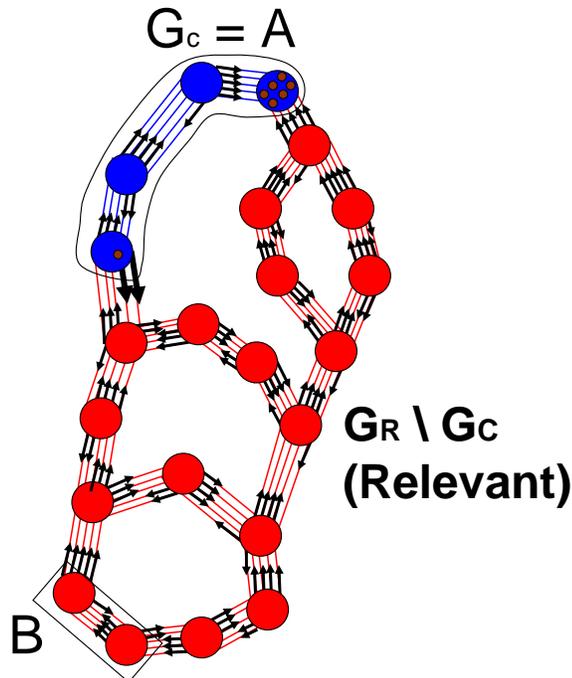


Figure 5.7: Example of allosteric type 3 communication (Increase of flexibility at a distance). The relevant region of the core $G_c = A$ includes the site B (see Algorithm 5.5.2), so removal of an edge in B increases the DOF count at A from one (internal) DOF (i.e. 7 free pebble) to two internal DOF (i.e. 8 free pebbles) and removal of two or more edges at B causes the core to have three internal DOF.

In the remaining examples we will primarily focus on allosteric type 2 communication.

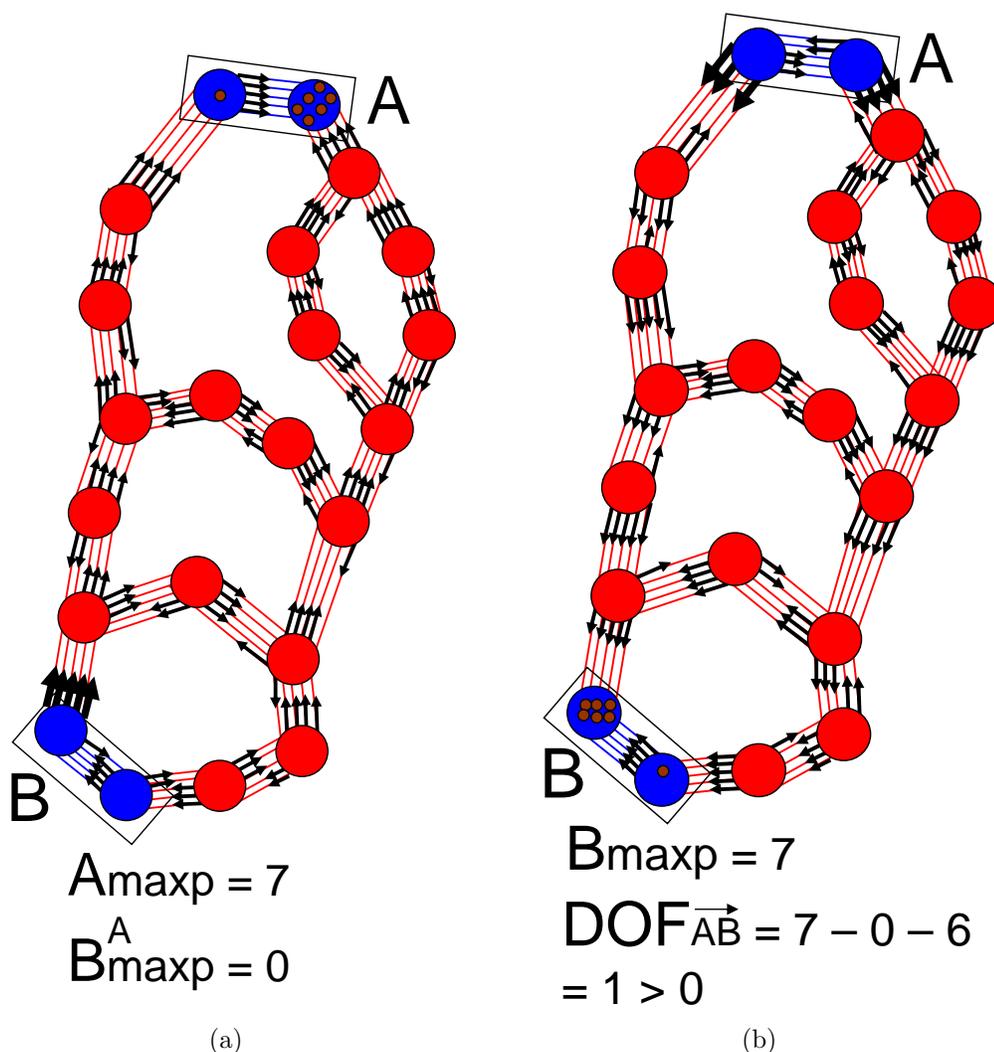


Figure 5.8: Illustration of allosteric type 2 communication and output of Algorithm 5.5.3. Rigidification of A causes a decrease of one DOF at B . The DOF that are transmitted from A to B by rigidification of A is $\text{DOF}_{AB} = 1$, and since it is positive, A and B are involved in allosteric type 2 communication.

Example 3 – Figure 5.8

We again revisit the physical model in Figure 5.5. We claimed that rigidifying site A would rigidify site B and is an example of allosteric type 2 communication, illustrating change in rigidity behaviour at a distance. To check for this, in Figure 5.8 we have shown the output of the allosteric type 2 Algorithm 5.5.3 (Transmission of DOF –

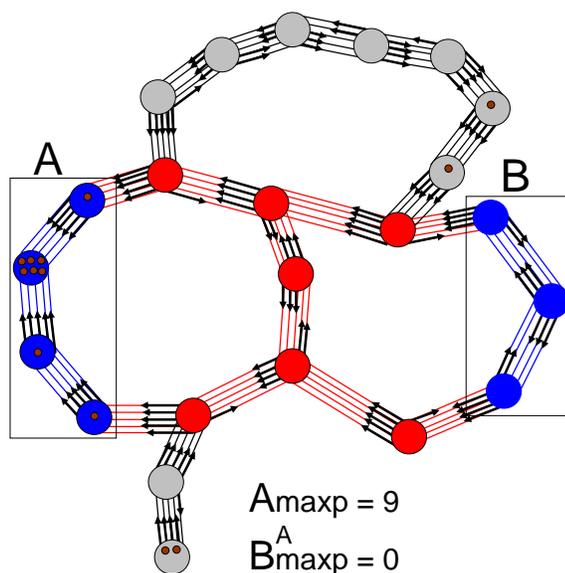
reduction of flexibility at a distance). In (a) we have played the pebble game on the entire graph and obtained the values of the counts $A_{maxp} = 7$ (maximum free pebbles on A - step 2) and $B_{maxp}^A = 0$ (maximum free pebbles on B while pebbles on A are frozen - step 4). In (b) we see that $B_{maxp} = 7$ (step 6). So it follows that $\text{DOF}_{\overrightarrow{AB}} = B_{maxp} - B_{maxp}^A - 6 = 7 - 0 - 6 = 1$. Since $\text{DOF}_{\overrightarrow{AB}}$ (i.e. maximum number degrees of freedom that can be transmitted from A to B , that is maximum number of DOF that can be removed at B by rigidifying A) is positive, sites A and B are involved in allosteric type 2 communication. Rigidifying A transmits a reduction of 1 (internal) DOF at B . The relevant region $G_R \setminus G_c$ of the core $G_c = A \cup B$, which is the remaining part of graph, is coloured in red, and corresponds to the region (allosteric communication pathway) over which allosteric type 2 communication (transmission of DOF) occurs ¹². ■

This example is an illustration of a simple transmission of DOF between two sites, as rigidification of one site causes a removal of the only remaining internal DOF from the whole structure, and therefore either site. As commented earlier, an insertion of a single independent edge makes this whole graph (structure) rigid. We now turn to slightly more complicated examples of allosteric type 2 communication.

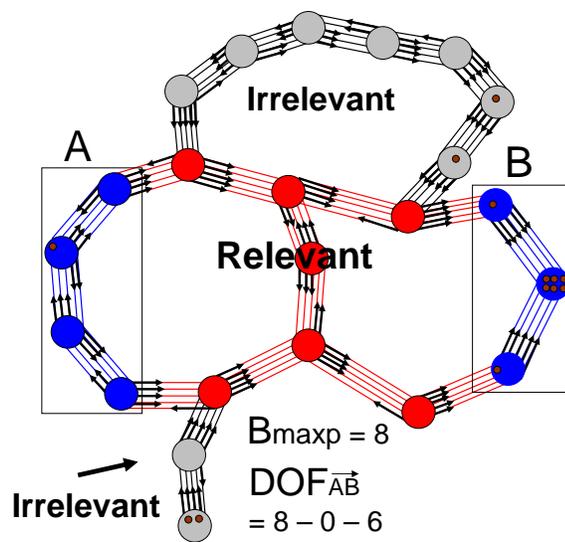
Example 4 – Figure 5.9

The site A in Figure 5.9 has 3 (internal) relative DOF (i.e. $A_{maxp} - 6 = 9 - 6$). Drawing back and freezing maximum free pebbles to A leaves no pebbles at B (i.e. $B_{maxp}^A = 0$) (a). Site B has 2 (internal) relative DOF (i.e. $B_{maxp}^A = 8$) (b). Therefore, rigidifying A removes 2 DOF at B (i.e. $\text{DOF}_{\overrightarrow{AB}} = 8 - 0 - 6 = 2$), so A and B are involved in allosteric type 2 communication. The relevant region with respect to the core $G_c = A \cup B$ is shown in red and irrelevant region in gray. Removal of

¹²Note that this count would be the same if we were to rigidify site B first (see below for generalization of this). That is $\text{DOF}_{\overrightarrow{BA}} = A_{maxp} - A_{maxp}^B - 6 = 7 - 0 - 6 = 1 = \text{DOF}_{\overrightarrow{AB}}$



(a)



(b)

Figure 5.9: Illustration of allostery type 2 communication. Rigidification of A causes a decrease of 2 DOF and rigidification at B . The DOF that are transmitted from A to B by rigidification of A is $DOF_{AB} = B_{maxp} - B_{maxp}^A - 6 = 8 - 0 - 6 = 2$, so A and B are involved in allostery type 2 communication. Note that we get the same answer if we rigidified B first. That is $DOF_{BA} = A_{maxp} - A_{maxp}^B - 6 = 9 - 1 - 6 = 2$.

irrelevant region (perfect bridge of length 8 and short dangling end) does not alter the rigidity-based allosteric type 2 behaviour of the two sites, and would not be part of the rigidity allosteric pathway.

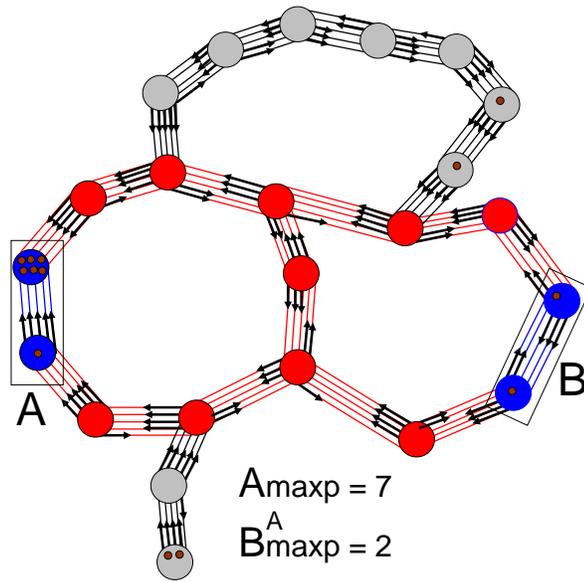
We note some features that were not seen in the previous example. First, the first independent edge that is added at A (i.e. one pebble removed) does not contribute to a decrease of free pebbles (DOF) at site B . The transmission only starts (i.e. removal of available free pebbles from B) once a second independent edge is added at A . When third independent edge is added, B is rigidified, that is $\text{DOF}_{\overrightarrow{AB}} = 2$ (internal) DOF (pebbles) are removed at B .

Let us now reverse the rolls and consider the effect of rigidification of site B on site A . Applying the Algorithm 5.5.3, we get $B_{maxp} = 8$, $A_{maxp}^B = 1$, $A_{maxp} = 9$, and therefore $\text{DOF}_{\overrightarrow{BA}} = 9 - 1 - 6 = 2$, which is the same transmission count as $\text{DOF}_{\overrightarrow{AB}}$. However, unlike in the case where A was rigidified first, addition of a first edge at B immediately removes a DOF at site A , and addition of the second (independent) edge at B , which rigidifies B , removes the second DOF at A . So, rigidification of site B still leaves one (internal) relative DOF at A . In other words rigidification of B (i.e. removal of two free pebbles) reduces the maximum free pebbles at A from 9 to 7. Any additional edges that are added at B would be redundant and would have no effect on the pebbles/DOF (motions) at site A . ■

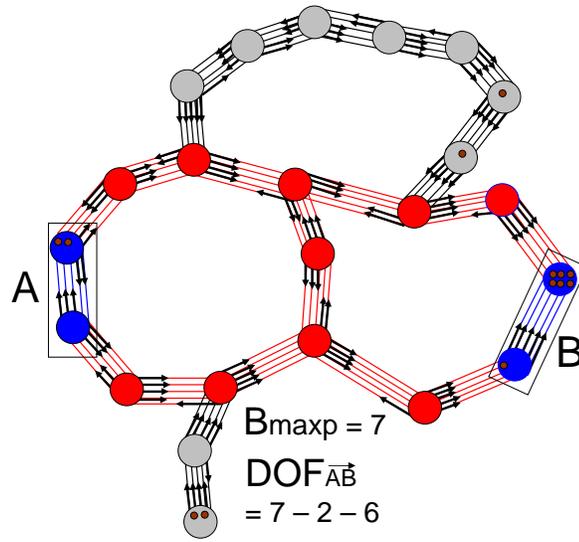
We note that Finbow-Singh and Whiteley have constructed some examples of allostery type 2 using special bar and joint frameworks [46].

Example 5– Figures 5.10 and 5.11

We now briefly show the output of the algorithm on two examples where the sites A and B are not involved in allostery type 2 communication. In Figure 5.10, rigidification of site A has no impact on the DOF at site B . That is rigidifying A , which is



(a)



(b)

Figure 5.10: Sites A and B are not involved in allosteric type 2 communication. That is rigidifying A has no impact on the DOF at B . Similarly rigidifying B has no impact on the DOF at A .

achieved by a removal of one free pebble at A , would still leave B with 7 free pebbles.

Applying Algorithm 5.5.3, we get that $A_{maxp} = 7$, $B_{maxp}^A = 2$, $B_{maxp} = 7$, so $DOF_{\overrightarrow{AB}}$

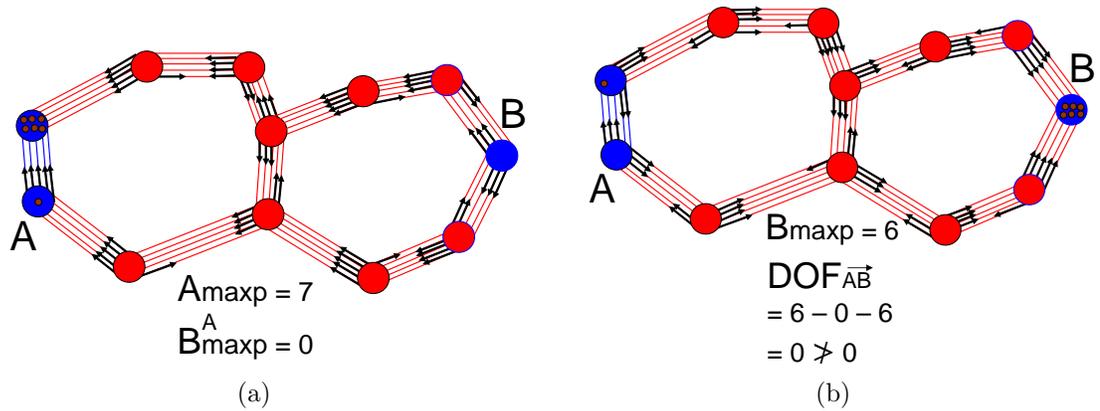


Figure 5.11: Allostery type 2 communication cannot occur when one of the sites is rigid. Addition of an edge in A has no effect on free pebbles in B .

$= 7 - 2 - 6 = -1 \not\geq 0$, confirming that A and B are not involved in allostery type 2 communication.

We recall, a basic assumption for allostery type 2 communication to be possible is that both sites A and B had to be flexible, that is they have positive internal DOF (i.e. maximum free pebbles at a site is greater than 6). In Figure 5.11, we have an example where one of the sites is rigid (site B), so algorithm correctly returns no transmission possible.

As a side comment, even though A and B are not involved in allostery type 2 communication in these two examples, note that any vertex in A and any vertex in B are still involved in allostery type 1 communication as there is a relevant region between them. ■

Example 6 – Figure 5.12

We now turn to a more complicated example of allostery type 2 interaction, where the answer is not trivial without doing the careful analysis and the run of the pebble game. We again apply Algorithm 5.5.3. We get that $A_{maxp} = 9$, $B_{maxp}^A = 1$, $B_{maxp} = 9$, and therefore $\text{DOF}_{\overrightarrow{AB}} = 9 - 1 - 6 = 2$. So rigidification of A removes 2 DOF

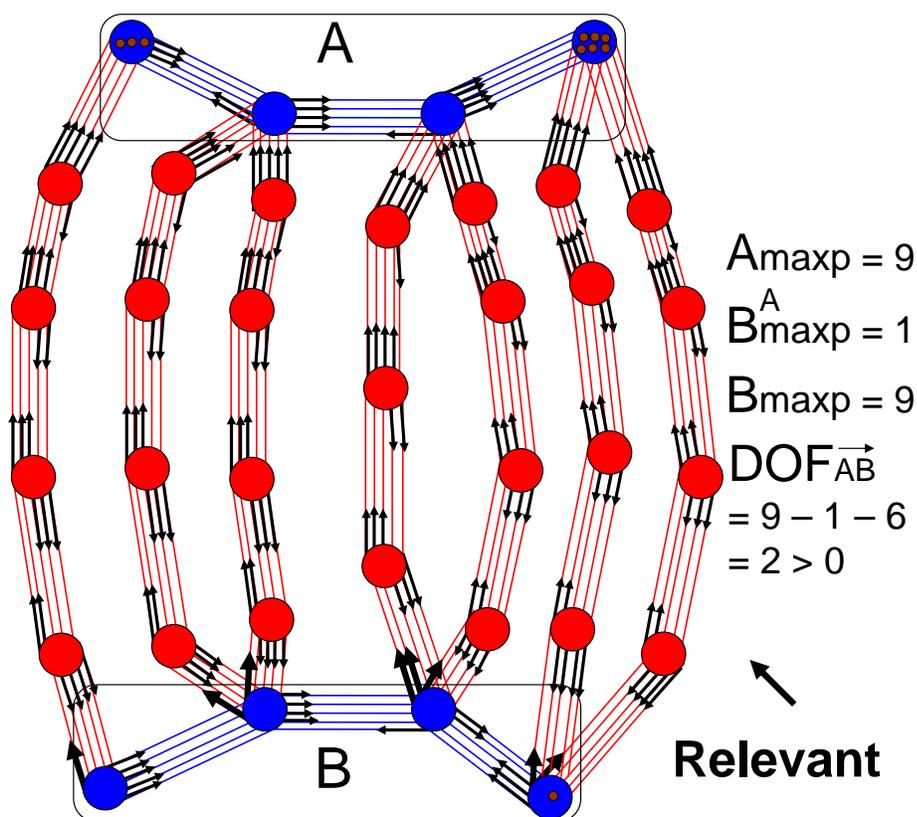


Figure 5.12: A and B are involved in allosteric type 2 communication, as the output of algorithm 5.5.3 tells us that that $DOF_{AB} = 2$ as, so rigidification of A removes 2 DOF from B . Note again it does not matter if B was rigidified first, we still get the same answer, $DOF_{BA} = 2$.

(pebbles) from B . Site A has 3 (internal) relative DOF ($A_{maxp} - 6 = 3$). The transmission of DOF only starts once second pebble is removed at A (i.e. second independent edge added) and once A is rigidified, one (internal) relative DOF still remains at B .

The relevant region of the core $G_c = A \cup B$ includes the rest of the graph and a removal of any edge(s) in the relevant region will modify the transmission count and/or stop the communication. The two sites are connected with 7 (perfect) bridges (see previous Chapter for definition), six are of length five and one located in the middle is of length four. If the bridge of length 4 is removed or if any two (or more)

bridges of length 5 are removed then there is no more transmission and A and B would no longer be in allosteric type 2 communication. Forming a new bridge of length 5 between A and B would cause an additional DOF to be transmitted (i.e. $\text{DOF}_{\overrightarrow{AB}}$ would increase by one), when rigidifying A . ■

In summary, the rigidity allosteric behaviour and shape transmission at a distance that we have observed in these examples convey several important messages. First of all, two distant sites can be connected in the sense of each three rigidity allosteric models where distant sites are involved in coordinated motions, increasing or decreasing flexibility at one site impacts the degrees of freedom (shape change) at the other site. The algorithms that were described in the previous section, extending the relevant regions detections algorithm and new pebble game procedures can efficiently predict these rigidity-based allosteric transmissions. We have seen some examples where rigidification of one site leads to a complete rigidification of the other site. The examples in Figure 5.9 and 5.12 illustrate when two sites are involved in allosteric type 2 communication, initial added edges (constraints) at one site may not immediately begin to transmit a reduction of DOF at the other site. In those cases, only once a certain number of constraints are added will transmission of reduction in DOF (change of shape) begin. Those examples also show that rigidification of one site, does not have to transmit to complete rigidification of the second site. Rigidifying one site, may still leave additional (internal) relative DOF at the second site, and any new edges added to the first site will only introduce redundancy and will never remove all the internal DOF from the second site (i.e. the second site will not be rigidified).

5.5.3 Properties of Allosterity type 2

As a reminder, both allosterity type 1 and 3 detection algorithms, essentially utilize the Relevant Regions detection algorithm, with certain modifications. Allosterity type 2 algorithm had additional algorithmic steps, and we made some observations earlier in this section. In this subsection we verify these previous observations and derive several other important properties about Algorithm 5.5.3.

In the first proposition, we assert that it does not matter how we have played the pebble game on the initial multigraph G (i.e. which order the edges in G were tested) and how we were drawing the pebbles to vertices in sites A and B , when we rigidify A we always get the same answer to how many degrees of freedom can be removed at B . In other words, the number of degrees of freedom transmitted from A to B is unique.

Proposition 5.5.4 *In the output of Allosterity type 2 algorithm 5.5.3, $\text{DOF}_{AB}^{\vec{}}$ is invariant under different plays of the pebble game.*

Proof The maximum number of free pebbles A_{maxp} that is initially recovered to vertices in A or maximum free pebbles B_{maxp} that is recovered to vertices in B (steps 2 and 6, respectively) is invariant under different plays of the pebble game (Theorem 3.3.2). In step 4 where we draw maximum number of free pebbles B_{maxp}^A to B , while keeping pebbles at A frozen, is also invariant as we simply continue to draw maximum free pebble to B , and we know that the maximum free pebbles to $A \cup B$ is also invariant. Since A_{maxp} , B_{maxp} and B_{maxp}^A are invariant, so is $\text{DOF}_{AB}^{\vec{}}$. ■

We now state an important theorem, confirming the previous claim and observations we made on examples, which says that the transmission of the maximum number of DOF from site A to B or B to A is identical. More specifically, the maximum degrees of freedom that can be transmitted from A to B , $\text{DOF}_{AB}^{\vec{}}$ (i.e. maximum

DOF that can be removed at B by rigidifying A) is equal to the maximum degrees of freedom that can be transmitted from B to A , $\text{DOF}_{\overrightarrow{BA}}$ (i.e. maximum DOF that can be removed at A by rigidifying B). So, it does not matter which site is rigidified (i.e. where we initially recover and freeze maximum free pebbles), the maximum DOF removed from the other site is not dependent on this choice.

Theorem 5.5.5 *Consider Allosteric type 2 algorithm 5.5.3. Maximum number of degrees of freedom that can be transmitted from A to B or B to A is the same, that is $\text{DOF}_{\overrightarrow{AB}} = \text{DOF}_{\overrightarrow{BA}}$.*

Proof Since, maximum number of free pebbles we can draw to the vertices of $A \cup B$ is invariant under different plays of the pebble game, then $A_{maxp} + B_{maxp}^A$ is invariant as well, which has the same number of maximum pebbles as $A \cup B$. In this pebble draw we first recovered free pebbles to A and then continued to B , but this still gives the same number of maximum free pebbles reversed. Similar argument applies to B_{maxp} and A_{maxp}^B , which also has same number of free pebbles as $A \cup B$. So,

$$A_{maxp} + B_{maxp}^A = B_{maxp} + A_{maxp}^B$$

Rearranging terms, we get that

$$\begin{aligned} & \text{DOF}_{\overrightarrow{AB}} + 6 \\ &= B_{maxp} - B_{maxp}^A \\ &= A_{maxp} - A_{maxp}^B \\ &= \text{DOF}_{\overrightarrow{BA}} + 6 \end{aligned}$$

So, we have that $\text{DOF}_{\overrightarrow{AB}} = \text{DOF}_{\overrightarrow{BA}}$ ■

Remark. Note that the $\text{DOF}_{\overrightarrow{AB}}$ count in algorithm 5.5.3 can also be obtained in the following alternate way. We only give the main counts that are needed without all the details. Let $G_c = A \cup B$ and denote the maximum number of free pebbles on G_c by $(AB)_{maxp}$. Then $\text{DOF}_{\overrightarrow{AB}} = A_{maxp} + B_{maxp} - (AB)_{maxp} - 6$. ■

We have made some remarks that both allostery type 2 and allostery type 3 are in some sense subclasses of allostery type 1. Specifically, when sites A and B are involved in allostery type 2 communication, then some pair of rigid clusters or vertices (one from A , other from B) defining two new sites are involved in allostery type 1 communication (i.e. coordinated motions). This is also true for allostery type 3 communication, as long as A and B are not part of the shared rigid region. It should be clear that the reason for this is that there must be a relevant region connecting the two selected sites.

We now show that allostery type 3 is a more specialized communication than allostery type 2, and under certain assumptions we can view allostery type 3 as a subclass of allostery type 2.

Theorem 5.5.6 *Assume A and B both have (internal) relative DOF (i.e. maximum pebbles recovered to A is greater than 6, likewise for B). Let core $G_c = B$, and assume that the relevant region of the core ($G_R \setminus G_c$) contains at least one pair of adjacent vertices (call them u and v) in A (i.e. A and B are involved in allostery type 3 communication) that are not part of a shared rigid region. Then A and B are involved in allostery type 2 communication.*

Proof To show that A and B are involved in allostery type 2 communication, it is enough to show that we can add one edge in A and that there is a reduction of DOF at site B (i.e. reduced number of maximum free pebbles). Assume that the maximum number of free pebbles is recovered to the core (site B). We know that in the relevant region $G_R \setminus G_c$ of the core there is no free pebbles and that the enlarged relevant region G_R has no outgoing edges in the pebble game generated directed graph. We add a new (independent) edge $e = \{u, v\}$ in A , removing a free pebble (DOF), and the seven pebbles that are placed on the ends of e (prior to it being pebbled) must be cascaded (reversed) out of the core (site B). So, an insertion of an edge in A has

reduced the maximum free pebbles on site B by one, so A and B are involved in allosteric type 2 communication. ■

Note that if A and B are involved in allosteric type 2 communication, they may not be involved in allosteric type 3 communication. For instance, see the example in Figure 5.8. Removal of any edge in either site, clearly has no impact on the maximum free pebbles/DOF at the other site.

In summary, we have seen examples of type 1 allosteric communication that do not have some special selection of sites (either the entire sites or some smaller selection within the two sites) that are in type 2 communication (see also Figure 5.14). Similarly, we have seen examples where two sites are involved in type 2 communication, but not in type 3 communication. So, in this sense allosteric type 3 is the strongest form of communication between the two sites. If we chose to allow for additional assumptions about the sites A and B (for instance in allosteric type 3, if we assumed both sites were flexible), then it would be possible to get a more stricter inclusion-wise relationship, where type 3 is a subtype of type 2.

5.5.4 Cyclic communication in Allosteric type 2

In the paper by Daily et al. [34] on protein allosteric communications, the authors formulated ‘the cyclic connectivity hypothesis’. In this hypothesis, they argue that in many allosteric proteins, mechanical allosteric coupling ‘... can occur only within cyclic substructures ... ’ of a protein. They further say: ‘In graph theoretic terms, the cyclic connectivity hypothesis entails that the allosterically connected subsets or “allosteric units” must be at least 2-connected ...’.

In the spirit of this hypothesis, in this section we prove that rigidity-based allosteric communication pathways of allosteric type 2 must be at least 2-connected, verifying the hypothesis of Daily et al.

Before we proceed, we will need a few simple graph theoretical definitions and some well known results about connected graphs. For any further terms and definitions refer back to Chapter 2 or see [38].

A graph G is *complete* if all its vertices are pairwise adjacent (i.e. connected by an edge). The complete graph on n vertices is denoted by K_n (i.e. K_3 is a triangle). Similarly, we can talk about a complete subgraph, where all vertices in the subgraph are pairwise adjacent. A graph G is *connected* if there is a path connecting every pair of vertices in G . Again, we can talk about subgraphs being connected. If $U \subseteq V(G)$ and $G(U)$ is connected, we say U is connected (in G). A maximally connected subgraph of G is called a *connected component* of G . So, a graph that is not connected can be divided into connected components. A connected graph has only one connected component. A *cut-vertex* is a vertex, such that when it is removed (along with all edges incident with it), produces a graph which has more connected components than the original graph (i.e. a cut-vertex disconnects the graph) (see Figure 5.13).

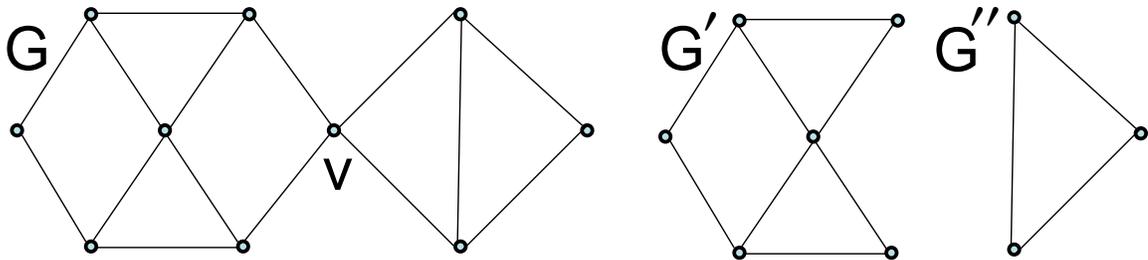


Figure 5.13: Graph G is a connected graph (has one connected component). G is not 2-connected as it has a cut-vertex v . When v is removed, along with its incident vertices, the two resulting subgraphs G' and G'' are both 2-connected. Both G' and G'' are blocks.

A graph G is said to be *k -connected* (for $k \in \mathbb{N}$) if $|G| > k$ and $G \setminus X$ is connected for every set $X \subseteq V$ with $|X| < k$ [38]. Less formally, G is said to be *k -connected* if there does not exist a set of $k - 1$ vertices whose removal disconnects

the graph. Note that a 1-connected graph is equivalent to a connected graph, and a 2-connected graph has no cut-vertex (see Figure 5.13). 2-connected graphs are also related by cycles: A graph (with at least three vertices) is 2-connected if and only if for all $v, w \in V(G)$ there exists a (simple) cycle C with $v, w \in V(C)$ [150]. One form of a famous Menger’s theorem in graph theory says: A graph (with at least $k + 1$ vertices) is k -connected if it contains at least k vertex-disjoint paths between any two vertices. So 2-connected graphs will have at least 2 vertex disjoint paths between all pairs of vertices [38]. A maximal connected subgraph without a cut-vertex is called a *block* [38]. So, every maximal 2-connected subgraph is a block. In a sense, blocks can be seen as the 2-connected analogue of connected components.

We first show an obvious statement which says that the relevant region of sites A and B in allosteric type 2 communication (which is true for other communications) forms a connected graph. As we are not making any assumption about connectivity of individual sites, we will first enforce connectivity of the sites by condensing the two sites to two individual vertices. Alternatively, as is done below, we can ensure connectivity of the individual site by constructing a complete graph on its vertex set. This will allow us to formulate a general statement about the connectivity, and subsequently 2-connectivity of the enlarged relevant region where the vocabulary will more closely depict Daily et al. “2-connected hypothesis” [34].

Lemma 5.5.7 *Assume A and B are involved in allosteric type 2 communication (i.e. $DOF_{AB}^{\vec{}} > 0$). Denote the enlarged relevant region as G_R (which includes G_c) of the core $G_c = A \cup B$. Condense subgraphs A and B to two new vertices A' and B' ¹³,*

¹³Briefly, without much technical ado, by condensation (also sometimes referred to as contraction [38]) of an induced subgraph, say A , we mean we (artificially) remove all the edges in A and contract (merge) all vertices in A into a single vertex A' , so that all the edges that had one endvertex in A now have A' as one endvertex. More detailed technical condensation/contraction procedure can be explained, but is not needed here, see [38] for instance. We said artificially remove edges, as it is important to note that we are not actually removing constraints and increasing flexibility (i.e. no extra free pebbles appear in the condensation of a subgraph). So, the newly formed vertices, A' and

denote the new core as $G'_c = A' \cup B'$ and denote new enlarged relevant region (with G_c replaced by G'_c) as G'_R . G'_R is a connected graph ¹⁴.

Proof Assume the maximum number of free pebbles are reversed to vertices in G_c . When A and B are involved in allosteric type 2 communication, in the pebble game generated directed graph there must exist some (capped) failed search region (i.e. relevant region) that starts from either A and reaches B and/or starts at B and reaches A . In other words, in terms of the graph with A and B condensed, there is a path linking (connecting) A' and B' . Furthermore, any vertices in $G'_R \setminus G'_c$ are found by (capped) failed searches out of G'_c (i.e. reachability region out of G'_c), which are trivially connected to G'_c . So, G'_R must be connected. ■

Proteins certainly have an underlying connected (molecular) graph. When there is more than one monomer, the interactions between the monomers ensure the connectivity of the larger oligomeric complex. The only reason we contracted sites A and B is because we allowed the possibility that one site (or both) may not be a connected subgraph. In other words, it is possible to pick some atoms (vertices) in a site where the induced subgraph by these vertices may not be connected.

We use the connectivity of the relevant region from the last statement to derive this important result that characterizes the underlying subgraphs of the allosteric pathways of rigidity-based type 2 allosteric communications.

Theorem 5.5.8 *Assume A and B are involved in allosteric type 2 communication (i.e. $DOF_{AB}^{\vec{}} > 0$). Denote the enlarged relevant region as G_R (which includes G_c) of the core $G_c = A \cup B$. There exists no cut-vertex in $V(G_R) \setminus V(G_c)$.*

Equivalently (by contrapositive): If there is a cut-vertex in $V(G_R) \setminus V(G_c)$ then A and B are not involved in allosteric type 2 communication.

B' , are not regular vertices in terms of the output of the pebble game as they can have more than 6 free pebbles and more than 6 outgoing edges in the pebble game generated directed graph.

¹⁴This statement also holds for types 1 and 3 communications.

Proof Assume for a contradiction that there is some cut-vertex v in $V(G_R) \setminus V(G_c)$. Any path linking (connecting) A and B in G_R must pass through v . If there exists another path that does not pass through v , then removal of v would not introduce more connected components to G_R (i.e. it would not disconnect G'_R). In any pebble game generated directed graph, the maximum number of outgoing edges out of any vertex v is 6, so the maximum number of pebbles that can be cascaded between A and B is 6 (where the cascade of pebbles must pass through v). However, this cannot lead to allosteric type 2, as we need more than 6 pebbles that can be cascaded (transmitted) between A and B . So, v cannot be a cut-vertex. ■

Note that an equivalent form of Theorem 5.5.8 tells us (using Menger's theorem): *If A and B are involved in allosteric type 2 communication there are (at least) two vertex disjoint paths¹⁵ connecting A and B .*

We can then immediately derive this corollary.

Corollary 5.5.9 *Assume A and B are involved in allosteric type 2 communication (i.e. $DOF_{AB}^{\vec{}} > 0$). Denote the enlarged relevant region as G_R (which includes G_c) of the core $G_c = A \cup B$. If there exists a rigid cluster in $G_R \setminus G_c$, whose removal increases the number of connected components in G_R , then A and B are not involved in allosteric type 2 communication.*

Proof The proof is identical to the proof of Theorem 5.5.8. We essentially treat the vertex v in that proof as a rigid cluster, and use the pebble game property that for any rigid cluster there are maximum 6 outgoing edges. ■

This Corollary can be very useful in practice. We can first find the relevant region of the core (two combined sites), and presence of any cut vertex or a larger rigid

¹⁵It is important to note that in this case the terminal (end and start) vertices (i.e. where they first enter/leave the two sites) of at least two paths connecting A and B will also be distinct. That is, the four terminal vertices of (at least) two paths between A and B are all different. This follows from the same reasoning given in the proof of Theorem 5.5.8, as it is not possible to transmit (receive) more than 6 pebbles from A to B through a single vertex.

cluster which acts as a cut-vertex in the relevant region, let us call it *cut-cluster*, would immediately indicate that we do not need to proceed with allosteric type 2 detection algorithm, as two sites cannot be involved in allosteric type 2 communication. In the protein applications, it often may be visually easier to locate a cut-cluster from the FIRST output than a single cut-vertex.

We now restate Theorem 5.5.8 in a special form using 2-connectivity, which captures the “cyclic, 2-connectivity hypothesis” of Daily et al. More specifically, this result asserts that allosteric communication pathways in allosteric type 2 communication between A and B are 2-connected. Since in this form of Theorem 5.5.8, we actually formalize the 2-connectivity of the enlarged relevant region G_R , we refresh our memory that we need to impose a connectivity condition of the two sites. This time, for each site we construct a complete graph on its vertex set (i.e. add an (artificial) edge between all pairs of vertices that are not adjacent in A , likewise for B). Denote the two sites with added (artificial) edges as K_A and K_B ¹⁶.

Theorem 5.5.10 *Assume A and B are involved in allosteric type 2 communication (i.e. $DOF_{AB}^{\vec{}} > 0$). Denote the enlarged relevant region as G_R (which includes G_c) of the core $G_c = A \cup B$. Let \widehat{G}_R be the enlarged relevant region G_R with artificial edges added to non-adjacent edges in site A and site B , creating complete graphs K_A and K_B , respectively. We denote $\widehat{G}_c = K_A \cup K_B$. \widehat{G}_R is a 2-connected graph.*

Proof \widehat{G}_R is connected (see Lemma 5.5.7) and K_A and K_B are connected by at least two vertex disjoint paths in \widehat{G}_R (Theorem 5.5.8). As there are clearly no cut-vertices in K_A and K_B and no cut-vertices in $V(\widehat{G}_R) \setminus V(\widehat{G}_c)$ which was shown in Theorem 5.5.8, then \widehat{G}_R is 2-connected. ■

¹⁶In the construction of the complete graphs K_A and K_B on sites A and B , it is important to keep in mind that we are not actually adding any additional constraints (removing pebbles in the pebble game output) to either A or B . The artificial edges are added so we can think of sites A and B as complete graphs.

In summary, in this section we have shown that the allosteric communication of type 2 must occur over a 2-connected subgraph.

We should mention that 2-connectivity, or more specifically lack of a cut-vertex (or cut-cluster) in the relevant region $A \cup B$, does not give us a sufficient condition for allosteric type 2 communication to occur. In Figure 5.14, we have shown an example where the relevant region (shown in red) of the core $G_c = A \cup B$ has two vertex disjoint paths connecting A and B (i.e. remove any red vertex and the graph is still connected), yet A and B are not involved in allosteric type 2 communication. The Algorithm 5.5.3 on this example tells us that $\text{DOF}_{\overrightarrow{AB}} \not\approx 0$ (i.e. rigidify one site, it has no effect on the DOF (pebbles) at the other site). The additional step in the Algorithm where we freeze the pebble on one site (i.e. site is rigidified) was critical in detecting when are two sites involved in allosteric type 2 communication.

Note that the example in Figure 5.14 serves another purpose. The existence of the relevant region between A and B suggests that the two vertices in A and two vertices in B corresponding to the ends of the two paths linking A and B (two pairs of vertices) are involved in allosteric type 1 communication, with 1 relative DOF between each pair. So, even without 2-connectivity, two sites can still communicate with each other (see below on Calmodulin). The three different types of rigidity-based allosteric models cover a larger variety of allosteric communications in proteins than type 2 alone would.

Note that the main result (Theorem 5.5.8) presented in this section on the necessity of 2-connectivity of allosteric type 2 communication does not hold for allosteric type 1. For instance, see examples in Figure 5.6 (a) and (b). In (a) removal of any vertex in the relevant region disconnects the two sites, and similarly in (b) removal of a vertex where two rings meet would disconnect the relevant region, yet the two sites are involved in type 1 communication. As we are proposing three types

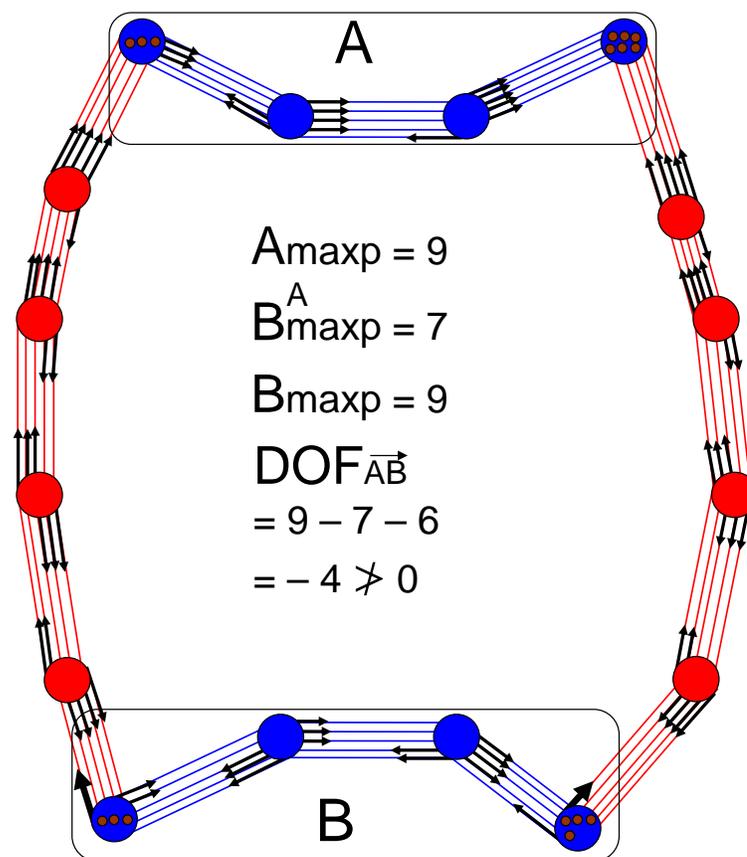


Figure 5.14: The enlarged relevant region of $G_c = A \cup B$ is 2-connected, yet the two sites A and B are not involved in allosteric type 2 communication.

of rigidity-based allosteric communication, we note that two sites can be involved in allosteric type 1 communication without having 2-connectivity in the relevant region (allosteric pathway).

We end the section by conjecturing that the general 2-connectivity results of the allosteric type 2 communication also holds for two sites that are involved in allosteric type 3 communication.

Conjecture 5.5.11 *Assume A and B are involved in allosteric type 3 communication, where removal of edge(s) in A increases pebbles (DOF) at B . Denote the enlarged relevant region as G_R (which includes G_c) of the core $G_c = B$. There exists no cut-vertex in $V(G_R) \setminus V(G_c)$.*

Equivalently (by contrapositive): If there is a cut-vertex in $V(G_R) \setminus V(G_c)$, then A and B are not involved in allosteric type 3 communication.

5.6 Applications to protein allostery

In this section, we move from the theoretical work and examples on small structures and graphs, and apply the rigidity-based allostery detection algorithms to actual protein structures. We will not give any detailed biological background on functions of any proteins that are discussed, as that would be distracting and beyond the scope of this work. We merely want to give a brief illustration of how the methods that we have proposed can be used in prediction of protein allostery. We will primarily focus our attention and illustrations on the implications of modelling rigidity allosteric interactions on an important class of transmembrane proteins called G-protein coupled receptors (GPCRs). We first demonstrate the rigidity-based allosteric communication ideas on the protein Calmodulin.

5.6.1 Rigidity-based allostery in calmodulin

At one time, examples of allostery were mostly limited to multimeric proteins (for instance hemoglobin), whereas today allostery is recognized as an important characteristic in monomeric proteins as well [68]. The allosteric behaviour of monomers with multiple binding sites is believed by biochemists to have the same underlying allosteric principles such as the long-range communication of sites, conformational

and rigidity/flexibility (shape) changes that also occur with large multimeric proteins (sometimes also referred to as oligomeric proteins or oligomeric complexes) [68].

As a side comment, we note briefly that there are some other important distinctions between multimeric and monomeric proteins, since in many multimeric proteins several copies of the individual protein chain assemble with symmetry, and symmetry of the structure may play an additional role in the allosteric communication [69] (see more general comments on symmetry and its impact on rigidity in the concluding section). To add to this, it is also not yet known if the timescales of conformational changes or changes in the flexibility involved in allostery are different depending on if the signal is sent within a monomer or through several subunit interfaces ¹⁷ [68].

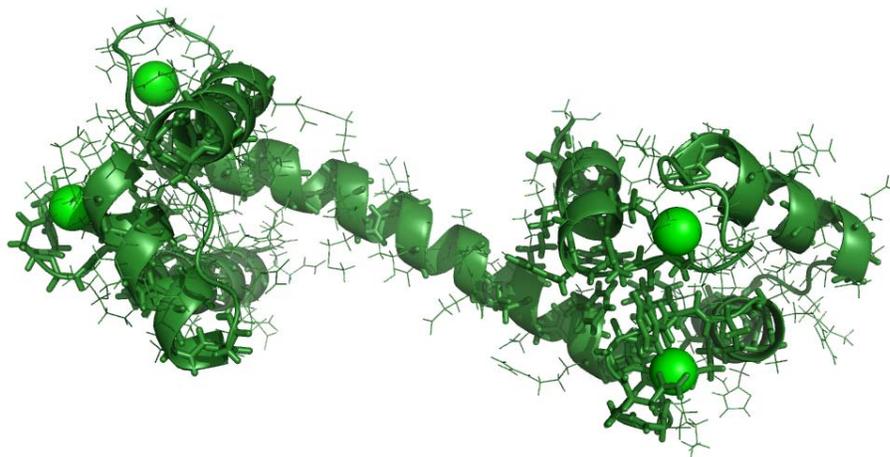


Figure 5.15: Three-dimensional structure of calmodulin with four calciums bound shown as green spheres (pdb id: 3cln). Recall that even though the central helix looks mostly straight, we have shown that there is a flexible hinge in the middle of the central helix.

In the previous chapter, we used our Hinge-detection algorithm to correctly predict the location of the hinge in Calmodulin, which is a monomeric protein (composed of a single polypeptide chain). To refresh our memory, in Figure 5.15 we display

¹⁷It is speculated that allosteric motions can occur on wide range of times scales including both fast motions occurring on pico- to nanosecond and slower motions on micro- to millisecond timescales [68].

the structure of calmodulin (with all four calcium bound), however, this time we have shown the locations of the four calcium-binding sites, with calcium depicted as green spheres. Out of the four binding sites, two are on the N-terminal half (1 and 2) and two on the C-terminal half (3 and 4), which are located on the opposite sides of the long central α -helix. We recall that the hinge was located roughly in the middle of the central helix, where the bending occurs (Figure 4.6 (b)).

In addition to undergoing a large hinge-bending motion, several studies have suggested that calmodulin also functions as an allosteric protein, although the allosteric mechanism in calmodulin is enigmatic and part of ongoing research [85, 106, 178, 217]. It has been proposed that the binding of calcium facilitates the transition between a low-affinity state and a high-affinity state [178]. When no calcium is bound, low-affinity state prevails, and when 2 or more calciums are bound, high-affinity state dominates (i.e. for instance it has been reported that there are nearly 200,000 calmodulin molecules in the low-affinity state for each calmodulin molecule in the high-affinity state) [106, 178]. Numerous studies report that calcium binding in calmodulin is highly (positively) cooperative (i.e. binding of calcium increases the likelihood (affinity) of the other binding sites to bind to calcium) [85, 106, 137, 178, 217]

¹⁸.

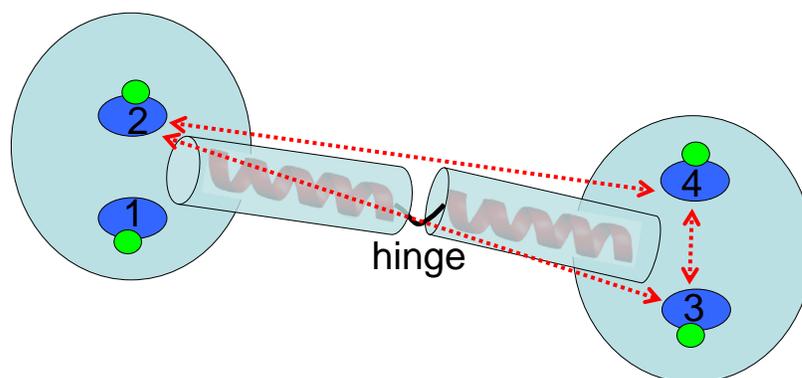
What makes calmodulin interesting and perhaps a part of a unique class of allosteric proteins is its unusual 3-dimensional structure, where the N-terminal binding sites and C-terminal binding sites are connected by a single α -helix. If the binding

¹⁸As a side comment, we should mention that it is generally difficult to find out the binding order as once site(s) become bound with calcium (typically 2 or more), the calcium binding affinity at other sites is tremendously increased, since the remaining sites could be binding simultaneously [211]. It would be interesting to know, for instance, does 1 first communicate with 2 (i.e. binding at 1 lead to binding at 2) then 2 communicates with 3 and 3 with 4 leading to stepwise loading of calcium?, or is there a simultaneous communication between the sites? and are perhaps any sites excluded in the communication (i.e. does binding of calcium at one site not impact the binding of calcium at another site(s)?) It has also been suggested that initial binding (i.e. initial loading of calcium at one site) may not impact the other sites, but once second calcium is bound there is cooperativity (see [85] and further comments below).

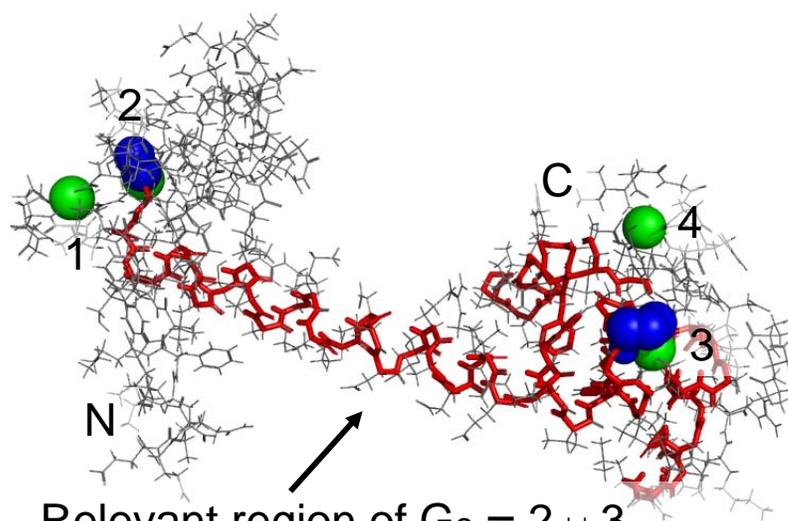
site(s) in the N-terminal half and binding site(s) in the C-terminal half are in allosteric communication, then the signal would have to be sent through a backbone chain composed of rigid clusters that are attached in a linear chain fashion. More specifically, the signal has to be transmitted through the central helix, which is composed of two large rigid clusters (see rigid cluster decomposition in Figure 4.6 (a)). So, the type of allosteric mechanism that is used in calmodulin must be somewhat unusual, which makes it a good case study.

Recall from the last section that 2-connectivity (i.e. lack of a cut-vertex or cut-cluster) in the allosteric communication pathway was the basic requirement for allosteric type 2 communication, which was also conjectured for allosteric type 3 (see Corollary 5.5.9, Theorem 5.5.10 and Conjecture 5.5.11 in section 5.5.4). The presence of a large rigid cluster (i.e. half of the central helix) between the N-terminal and C-terminal sites suggests that allosteric type 2 and 3 are not possible between the N-terminal and C-terminal sites. So, if the rigidity-based allosteric communication is involved between the N and C-terminal sites then the only possibility is that it is allosteric type 1.

To investigate this further, we performed our rigidity-based allosteric analysis (essentially Algorithm 5.5.1) as described in the previous section on the calcium bound calmodulin structure (pdb: 3cln). We analyzed all possible pair-wise communications among the 4 binding sites (i.e six pairs). The sites were defined as atoms (vertices) which are in the immediate proximity to the calcium ligands. The residues that contain these atoms were checked on Catalytic Site Atlas [141] and in literature [217] to be part of calcium binding sites. More specifically, to check for rigidity-based allosteric communication, we have defined the core to be two of the sites and then we searched for a relevant region connecting the two sites.



(a)



Relevant region of $G_c = 2 \cup 3$
(allosteric pathway)

(b)

Figure 5.16: Binding sites 2, 3 and 4 in calmodulin (calcium bound state - pdb id 3cln) are involved in rigidity-based communication (allostery type 1 - coordinated motions) with each other (a). No site is communication with site 1. The red arrows indicate that the two sites are in communication with each other. In (b) we have shown the relevant region (shown in red) of sites 2 and 3 (shown in blue), the calciums are shown in green. The relevant region is the region in the protein that is important for this communication (i.e. rigidity-based allosteric pathway) between sites 2 and 3. Note how the relevant region extends all the way from one binding site across the central helix to the other binding site. The gray region in the protein is irrelevant and not important for this communication.

The first thing that we observed is that no site is in communication with site 1 (i.e. no relevant region connecting the two sites). In other words, freezing site 1, had no affect on the motions (DOF) of the other sites (i.e no site is in a coordinated motion with site 1). On the other hand, we find that all the other combinations of pair of sites are involved in allostery type 1 communication as there is a relevant region connecting the two sites (i.e. 2 and 3, 2 and 4 and between 3 and 4). So, motions of sites 2, 3 and 4 are coordinated with each other, while site 1 seems to be excluded from the site pairwise coordinated motions (see Figure 5.16 (a)).

In Figure 5.16 (b) we have displayed the output of the relevant region of sites 2 and 3. The sites are shown in blue spheres and calciums as green spheres, and the relevant region connecting the two sites is displayed in the red with the stick representation. The relevant region corresponds to the region that is important for the communication ¹⁹ between the two sites (and can be thought of as the allosteric pathway). Note how the relevant region extends from one binding site all the way to the other binding site.

In the view of the tug-of-war analogy that we have provided earlier, the relevant region of the two sites has constrained the two sites with respect to each other the appropriate amount (i.e. rope is taut enough), which makes the coordinated motion of the pair of distant sites possible. One site is not freely moving relative to the other, as its DOF (motions) are restricted by the other site. For instance, a pair of atoms one from site 2 and one from site 3 have 4 relative DOF between them (i.e. freeze one, the other now only has 4 DOF). In terms of the basic properties of the relevant regions, removal of any constraints (bonds, edges) in the relevant region can significantly impact the communication between the two sites and very likely

¹⁹Instead of saying that two sites are involved in rigidity-based allosteric communication, for brevity we may sometimes just say that two sites are in ‘communication’ or ‘communicating’ with each other.

completely cease the communication. On the other hand, removal of any constraints in the irrelevant (gray parts of the protein) will have no impact on this communication. These findings can perhaps have important implications in the mutation studies, as mutations in the relevant region of the two sites are likely to affect the communication (coordinated motions) of the two sites and possibly their ability to cooperatively bind to calcium. For instance, if the hinge region were to become more flexible ²⁰, then the communication (coordinated motion) between the two sites will likely not be possible.

Similar reasoning and comments would apply to the relevant regions between sites 2 and 4 and between sites 3 and 4. Since site 3 is part of a rigid cluster, the communication between sites 3 and 4 cannot be of type 2 (i.e. in type 2 both sites have to be flexible), and it has to be type 1 allosteric communication.

To make the results more robust, we also ran the relevant region analysis by declaring the core to be one of the atoms from the clusters of atoms in one site and another atom from atoms in the second site. We found that it did not matter which two vertices (atoms) (one from one site and one from the other) was used to define the core, each choice would always find a relevant region between the two sites and no relevant region with site 1. This indicates that the results are not sensitive to these choices, so if another user chooses a different set of atoms in the binding sites as part of the core, the declaration of which sites are communicating should be the same (i.e. sites 2, 3 and 4).

For completeness and comparison purposes, we also ran the same procedure on the apo-structure (calcium free) of calmodulin (pdb id: 1cfd) and we found that no

²⁰Imagine that some residues in and near the hinge are replaced with prolines. This may introduce additional flexibility, due to the inability of prolines to contribute to the backbone hydrogen bonding that is needed for the helical rigidity. The backbone amide hydrogen in prolines is lost in the formation of a five member ring which initially adds more rigidity (due to a loss of a rotational DOF of backbone C_{α} -C bond), but with a loss of backbone hydrogen bonding, the net effect will likely cause an increase in DOF. Prolines preclude normal helix geometry, and are frequently found in flexible protein turns [15].)

sites were involved in a coordinated motion. This finding might suggest that having (some) sites in the ligand (calcium) bound conformation enables further the allosteric communication and subsequent (cooperative) increase in affinity to calcium binding, which is in general agreement with the previous studies [106, 178].

Hu et al. [85] have suggested that “loading the initial calcium (they do not mention which site) has little affect over the rest of the sites” and once the second calcium is bound, “it induces structural changes that cooperatively enhance the affinity of the other sites”. Assuming calmodulin binding can be explained by these events, it may explain why we found one site (i.e. site 1) was not communicating (i.e. was not in coordinated motion) with the other sites, meanwhile all the other three sites were involved in rigidity-based allosteric communication. Furthermore, the calcium binding sites are nonidentical [178] (i.e. do not have exact same residues), and are not symmetrically related, so it might not be as surprising why we have found one site was excluded in the communication. Moreover, our prediction of communication between sites 3 and 4 in the C-terminal is consistent with previous studies that find these two sites to be highly cooperative [85, 137], whereas sites 1 and 2 in the N-terminal are not cooperative [137]. Sites 3 and 4 are also reported to have much higher calcium binding affinities than the N-terminal sites 1 and 2 [85, 137, 217].

In summary, given that we are able to detect allosteric communication (i.e. coordinated motion) between sites 2 and 3 and between sites 2 and 4 is the most striking prediction as these pairs of sites are widely separated. Furthermore, the signal is able to transmit through the rigid clusters in the central helix, suggesting that in terms of our rigidity-based allostery models, only allostery type 1 communication could be used. Allostery in calmodulin has been previously studied, with suggestions that extra rigidity in the hinge motion (in the calcium bound state) may play an important role [106], but the mechanism of allosteric communication in calmodulin,

to our best knowledge, is not known. The presence of a relevant region between distant binding sites in the calmodulin protein, points to our proposed rigidity-based allosteric communication as a possible allosteric communication mechanism. These findings reinforce the power of a rather simple pebble game procedures and our extensions for detecting rigidity-based allosteric communication.

We now turn to an application of rigidity-based allosteric modelling to an important class of transmembrane receptor proteins.

5.6.2 Applications to G-Protein Coupled Receptors

Cells receive information from the external environment via proteins called *receptors* which span the membrane [3]. Some receptors are able to transfer the information through membrane by ion channels that enables ions to enter the cell [15]. However, most receptors are not involved in material transfer [135]. Instead, majority of transmembrane receptors utilize allosteric communication, where the mechanical effect of ligand binding (such as hormone or drug) on the extracellular part of the receptor causes the conformational and shape change, which travels to the intracellular (cytoplasmic) region of the receptor leading to a cascade of intracellular responses [21, 26, 135, 215]. Finding a mechanism by which transmembrane receptors propagate a signal across the membrane is a highly important problem in biology and in medicine.

A majority of transmembrane receptors are composed of several α -helices (sometimes referred to as transmembrane helices), which traverse through the membrane often in a nearly parallel fashion (like a set of long and narrow cylinders). The most common transmembrane receptors and the largest class of receptors in the human genome are the family of proteins called G-protein-coupled receptors (GPCRs)

[15, 26]. GPCRs mediate most transmembrane signal transduction²¹ across the cellular membrane by responding (binding) to an enormous variety of extracellular stimuli: small molecules such as drugs, hormones, neurotransmitters, proteins, to name some [21]. They also mediate the perception of light, taste and smell [3]. When in the active state, they further activate (primarily) their cognate G proteins, which triggers a cascade of intracellular responses [21]. Since GPCRs are responsible for the control of most information that passes into the cell from external environment, they play a critical role in many diseases (i.e heart disease, hypertension, cancer, asthma, diabetes, inflammation and psychological disorders, etc. [3]). Naturally, these proteins are the most commonly targeted receptor class (i.e. cellular docking stations) for medicinal therapeutics. In fact, it has been estimated that GPCRs are targets to more than 50% of all modern medicinal drugs given to patients, yet remarkably their activation mechanism is still largely elusive and not fully understood [215].

This large family of signalling proteins is characterized by a common topology consisting of a bundle of 7 transmembrane(TM) helices connected by extra- and intracellular loops²². In Figure 5.17 we have displayed a typical structure of a GPCR²³. The phenomenon of allostery is prominently seen in GPCR [3, 112, 181, 215]. GPCRs are activated when an activating ligand (commonly called *agonist*)²⁴ binds at the extracellular binding region, causing small shape changes in the ligand binding pocket where it interacts with the helices. This ligand-binding shape change drives through

²¹Signal transduction in biology, roughly speaking, is a process where the binding of extracellular signalling molecules and ligands to membrane receptors initiates events inside the cell.

²²Another common name for GPCRs is a *seven-transmembrane domain receptors* or *heptahelical receptors* due to the presence of 7 transmembrane helices

²³Note that some transmembrane receptors have different number of helices than GPCRs, for instance aspartate receptor has two [63]

²⁴In pharmacology, the activating ligand of GPCRs is commonly referred to as *agonist* (from the Greek 'contentant') (although other more detailed names and classifications are used [26]), and the inactivating ligand as *antagonist*. When the antagonist ligand binds it blocks the action (binding) of agonists (either directly or allosterically) keeping the receptor in inactive resting state. The agonist and antagonists can be endogenous (such as hormones and neurotransmitters) or exogenous (such as drugs) [26].

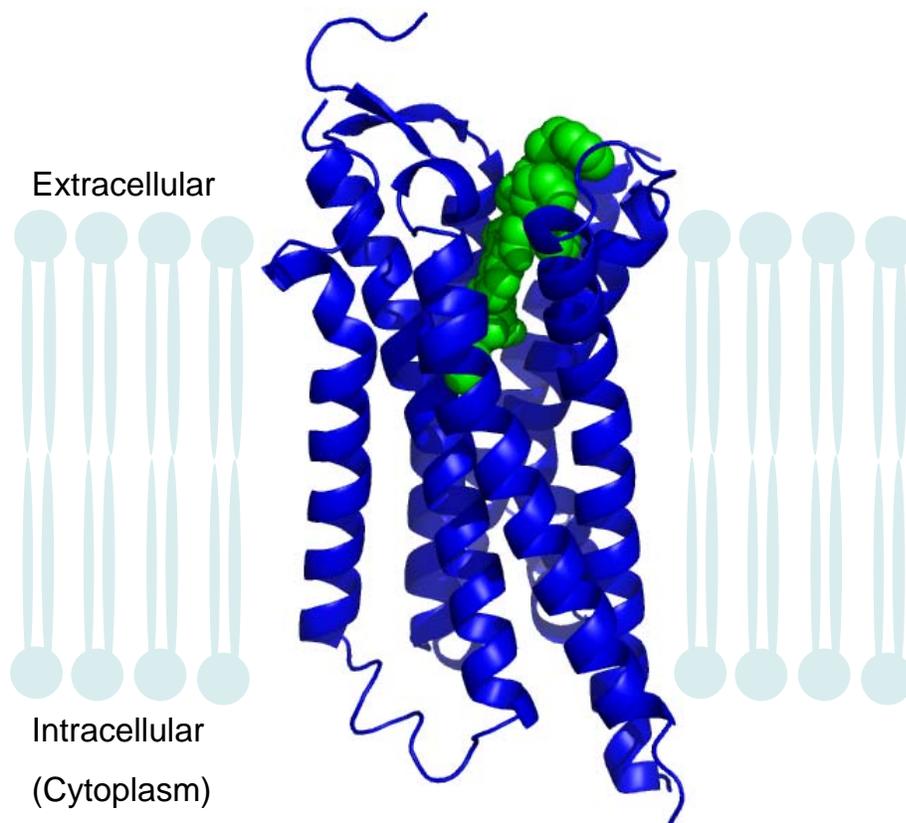


Figure 5.17: A typical structure of a GPCR. The seven transmembrane helices are almost parallel with respect to one another. Here we have shown the recently solved structure of an agonist-bound A_{2A} human adenosine receptor (pdb: 3qak), also known as the caffeine receptor. The protein is crystalized with the bound drug (shown in green spheres), that is a potential drug used for treatment of chronic obstructive pulmonary disease. When this ligand binds to the adenosine receptor, this causes the seven helices to shift and slide relative to one another [215]. Generated with Pymol.

the transmembrane helices which causes further shape changes to occur at the intracellular (G-protein) binding regions, leading to functionally important changes (such as activation of a G-protein [3, 113, 190, 215]²⁵). As we do not need detailed biological descriptions here, the main message that is needed for our purposes is that this

²⁵We are only giving some basic biology background for motivational purposes. Our goal here is not to be distracted by discussion on detailed biological processes and functions of the proteins discussed, for details see [3, 15, 26] for instance.

important class of protein are natural allosteric proteins, involving communication (shape changes) between two topologically separated sites.

The mechanism of allosteric transmembrane signalling still remains largely unsolved [112, 113]. It has been speculated that slight (coordinated) movements and rearrangements of transmembrane helices (upon ligand binding) are important for this allosteric communication [215].

Although still in the infancy, the very recent (last several years) and exciting breakthroughs in the determination of crystal structures of GPCRs has resulted in a few solved GPCRs [21, 113, 215]²⁶. These initial solved GPCRs, provide new opportunities for probing their allosteric mechanism. All GPCRs are structurally very similar, and as even more GPCRs are solved, it will bring about better understanding of molecular recognition and signalling and will aid in further drug discovery and development of more specific drugs [21, 26, 113, 215].

GPCR's may use a piston-type motion to transmit information

We give a brief review of one possible allosteric mechanism which has been hypothesized in a number of studies over the last decade and a half. We will not investigate this hypothesis, it is merely given for background purposes, with some comments and connections in the concluding section.

In this hypothesis, the transmembrane helical receptors, such as the GPCRs, would propagate the necessary conformational changes across the membrane using motions which closely resemble a sliding, piston-type of a motion [22, 23, 44, 63, 135, 194]. In the piston-motion hypothesis, the conformational changes initiated by a drug or ligand binding at the extracellular side of the receptor, are presumed to be transmitted to the intracellular regions by very small sliding motions of some helices

²⁶Compare to soluble proteins, it has been very difficult to experimentally obtain a 3-dimensional structure of a membrane protein.

[63, 135]. The sliding (shear) motions of helices relative to one another is thought to avoid collisions between helices and it allows helices to maintain their well-packed interface throughout the motion [63]. The piston-type motion hypothesis still remains largely unverified, primarily due to a very low number of solved transmembrane helical structures, particularly in both active and inactive states.

This hypothesis suggests that the helices that are involved in piston motions are sliding with respect to each other like a long bundle of cylinders. To facilitate these motions, the individual helices are moving like rigid objects with respect to each other. This is expected, as sufficiently long helices (with good hydrogen bonding geometry) such as the transmembrane helices, should have enough constraints (hydrogen bonds) to be rigid (in fact many are redundantly rigid, see [207] on confirmation of this using rigidity-based counting arguments). Moreover, helices are reported to be highly stable and rigid when embedded in the membrane [216].

To get a better visual sense of this hypothesis, in Figure 5.18 we have schematically shown 7 (transmembrane) helices as they occur in GPCRs. Helices are depicted with long cylinders and the ligand (depicted with purple) binds between two of the helices (cylinders). The arrows on the green cylinders indicate that these helices are undergoing relative piston-like sliding motions with respect to one another. In the right part of the figure we have displaced the green helices. Note that we have exaggerated the amount of sliding/shearing of the cylinders in this schematic. In actual proteins the sliding motions of helices are very small, typically on the order of just 1 or 2 angstroms [63, 135].

In the literature the piston motion hypothesis is very loosely defined. Even though the nomenclature of 'piston-type motion' and 'relative sliding' of helices suggests pure translational (shear) motions of helices along its axis, however, other motions for instance with rotational components are also suggested [63, 215], which likely

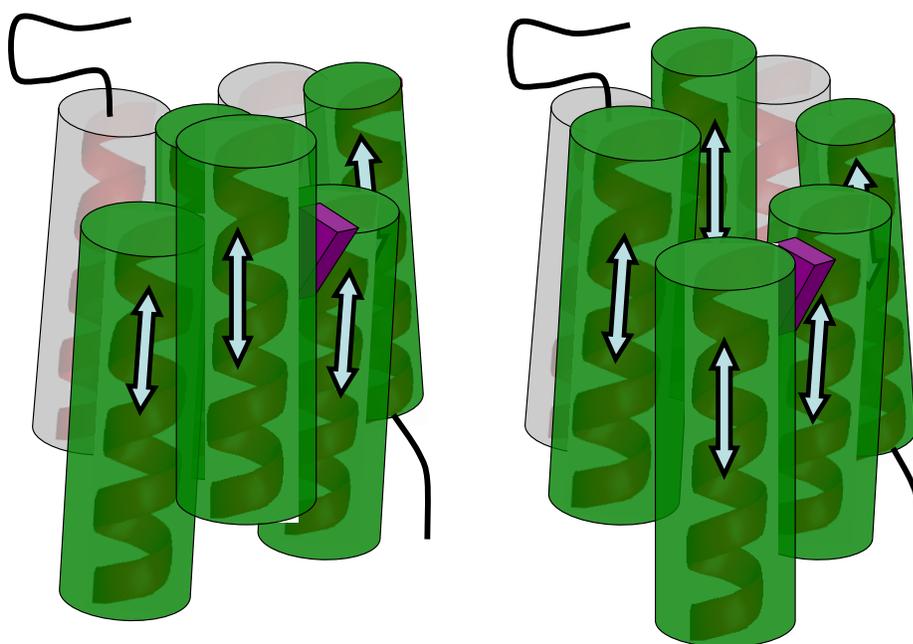


Figure 5.18: A schematic representation of the hypothesized piston-type motion mechanism that is speculated to transmit changes in shape across the membrane from the extracellular part of the GPCRs to the intracellular parts. In this hypothetical example, 5 out of the 7 (transmembrane) helices (depicted as green cylinders) are undergoing relative sliding motions with one another, and depicted to move up and down with respect to each other. The purple object is supposed to depict a ligand, which normally binds between two or more helices.

resembles a screw motion (i.e. translation and rotation) down the helical axis. We should also remember that this is only a hypothesized allosteric mechanism, and as with most allosteric proteins, there could be other mechanisms that are involved in shape transmission changes [215].

As mentioned earlier, we are not going to investigate the possibilities of the piston-type or other motions that could be used in the GPCRs, as our combinatorial algorithms and any combinatorial pebble game procedures that we have developed are not designed to test for the type of geometric (finite) motions. Furthermore, there are geometry reasons why sliding-like motions are preferred (i.e. for instance they more likely avoid collisions), which are not accessible to any tools, such as the pebble game

algorithm, that just track combinatorial (counting) properties of rigidity. To test out for the possibility of the piston type motions requires geometric analysis and simulations of (finite) motions (i.e. with FRODA, MD simulations etc.). We only wanted to briefly introduce this hypothesized allosteric mechanism in transmembrane helical receptors for general background purposes. We will make a few general connections and comments in the concluding section on the possible next steps and extensions of our work which should be investigated further in the future work.

Toy model of rigidity-based communication across a bundle of helices

Before we get into specific applications of our rigidity-based approach to allostery on GPCRs, let us briefly consider one toy model, demonstrating how rigidity-based allosteric communication between two widely separated sites can occur through a graph that closely depicts a structure of a transmembrane helical receptor, with a bundle of helices.

In Figure 5.19 we have displayed a multigraph that contains 4 hypothetical transmembrane helices, where each helix is assumed to be rigid (see below for more), and can be thought of as a single vertex (with 6 trivial rigid body motions).

In this example, we can see all three types of rigidity-based allosteric communications. Rigidification of one site rigidifies the other site (with 1 DOF being transmitted). Furthermore, a removal of any edge(s) in B increases the DOF count at A . The relevant region of the two sites (i.e. allosteric communication pathway) goes over the entire graph and through four helices reaching both sites. So, in this example we see that the rigidity-based allosteric communication involves all four helices. We should also note that every possible pair of helices are in a coordinated motion with one relative DOF between them. So, movements of any helix are coordinated with movements of other helices. The significance of this will be discussed

later. Any changes in the relevant region (i.e. addition/removal of edges) will modify the communication between the two sites, and change the nature (i.e. relative DOF) of coordinated motions between the helices.

Now we will apply our rigidity allostery procedures to a few recently solved crystal structures of GPCRs.

Examples of rigidity-based allosteric communication in GPCRs

Our main goal here is to provide evidence for the existence of rigidity-based allosteric communication between two widely separated sites in GPCRs. We will initially only test for type 1 communication, as it is the necessary condition for the other types. So, we will first look for a relevant region between some distant sites. When there exists such a relevant region, as a short form for rigidity-based allosteric communication, we will sometimes simply refer to it as communication between two sites. One site will typically be selected to be the ligand binding pocket or the ligand itself (which is located at the extracellular end of the protein) and the other site is in the intracellular region of the protein (where G proteins are known to bind [215]). We will end this subsection with explicit illustration of type 2 and type 3 communication.

We begin with application of our methods to a solved structure of the Human beta-2 adrenergic receptor (pdb id: 2rh1). These receptors are biologically very important, as they are responsible for the response to physical stress and play important roles in regulating cardiovascular and pulmonary function [21]. They are also implicated in multiple medical conditions such as cardiac arrhythmia, hypertension and bronchial asthma, and is a major target for treatment of hypertension and arrhythmia, with well known beta-blocker drugs [21].

Remark. We note that FIRST was mainly designed and fine tuned to handle and predict rigidity and flexibility of soluble proteins. In one previous study [145], FIRST

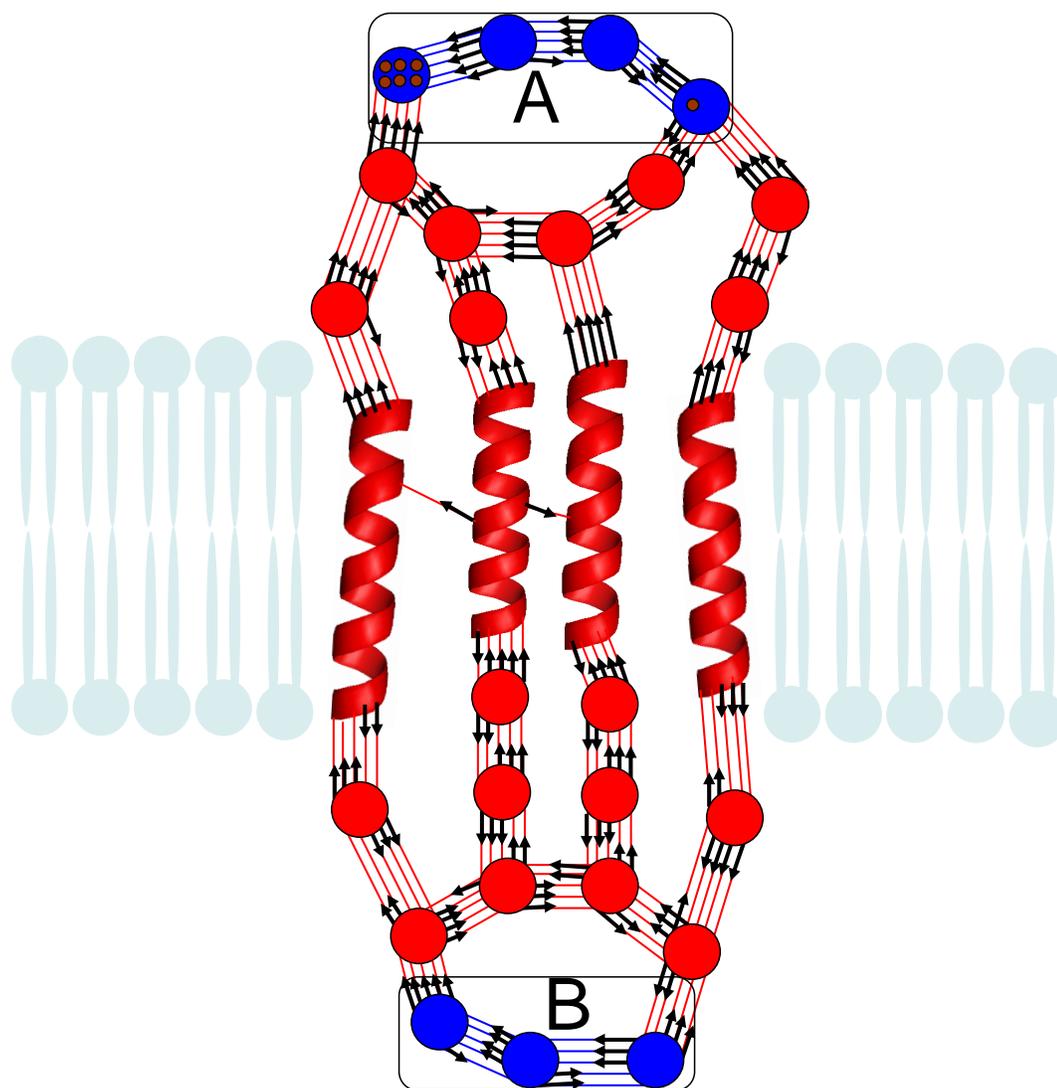


Figure 5.19: A hypothetical scenario that could occur in allosteric communication of two sites across transmembrane helices. Helices are modelled as rigid objects and can be treated as a single vertex. The pebble game output is displayed, with one internal DOF remaining in the graph. Sites *A* and *B* are involved in all three types of rigidity-based allosteric communication, and the relevant region (i.e. rigidity-based allosteric pathway) goes through all four helices. Rigidifying one site rigidifies the other site (remember which site is rigidified is not important, same effect is obtained). Removal of any edge(s) in *B*, increases DOF at *A*. Vertices (atoms) from one site are in coordinated motions with vertices (atoms) in the other site. Any pair of helices is in a coordinated motion.

was successfully applied on a GPCR rhodopsin without any consideration of the membrane lipid molecules, where the predictions agreed with experimental results.

Another study [99] also used FIRST to study rigidity and flexibility of a membrane protein. Our FIRST predictions on GPCRs (given below) in this section are a good first approximation of rigidity where several features are in agreement with experimental observations (see below). We also mention that we have done some initial checks using FRODA simulations on GPCRs, and we found that helices were not unravelling and remained relatively tightly packed and close to each other. Nevertheless, since the number of available solved membrane protein structures is rising, in future research we believe that further investigations and tests should be made and perhaps some adjustments and tweaks in hydrogen bond cutoffs or hydrophobic tethers should be incorporated into FIRST so it can handle general types of membrane proteins. ■

In Figure 5.20 we show the FIRST rigid cluster decomposition of Human beta-2 adrenergic receptor ²⁷. We observe that FIRST identifies most α -helices as separate rigid clusters, with one exception, where two helices form a single rigid cluster. We have labelled the rigid clusters of transmembrane helices, with rigid cluster 5 containing two helices. As a convention, rigid cluster helix 1 is at the N-terminus, and rigid cluster 6 towards the C-terminus of the protein. The ligand is now shown in orange spheres.

To perform the allostery analysis, we have selected the ligand as one site (which penetrates the cavity between the transmembrane helices) and part of the intracellular region as another site, and declared them as core (see Figure 5.21). We have depicted these two sites with blue spheres. When we apply the relevant region detection algorithm, we find that these two widely separated sites are communicating

²⁷In all the FIRST runs in this section on the GPCRs, the hydrogen bond energy cutoff in FIRST was selected at the earliest point where the protein decomposes into multiple rigid clusters (which typically is around the recommended cutoff of $E = -1$ kcal/mol (see FIRST manual for details [48] or discussions in the first or next chapter), which in most cases represents distinct (or the highest number that FIRST can identify) α -helices as rigid clusters.

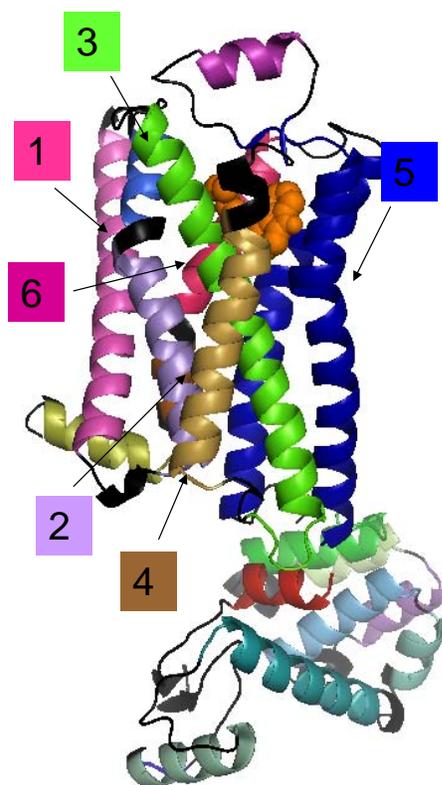


Figure 5.20: FIRST rigid cluster decomposition of Human beta-2 adrenergic receptor (pdb id: 2rh1). Most transmembrane α -helices are decomposed into separate rigid clusters, with rigid cluster 5 being the exception which is made of two helices. The ligand (located at the extracellular end of the protein) between the bundle of helices is shown in orange spheres.

with each other, as there is a relevant region connecting the two sites. The relevant region is shown in red, and the rest of the protein (i.e. the irrelevant region) in gray. So freezing one site, reduces the possible motions (DOF) (i.e. maximum pebbles) at the other site, so their motions are coordinated. We note that the relevant region (rigidity-based allosteric communication pathway) passes through most transmembrane helices, except helix 1. The helices that are part of the relevant region are constraining (reducing) the possible motions (DOF) of the two sites, and if the two sites were artificially rigidified into one rigid cluster, the relevant region would become part of the same rigid cluster. The N-terminal helix seems to be the most separated

in space from the ligand binding site ²⁸, and in addition to not reducing the relative DOF of the two sites (as it is not part of the relevant region) it is likely that it is not significantly constraining the motions of the other helices (see below for more).

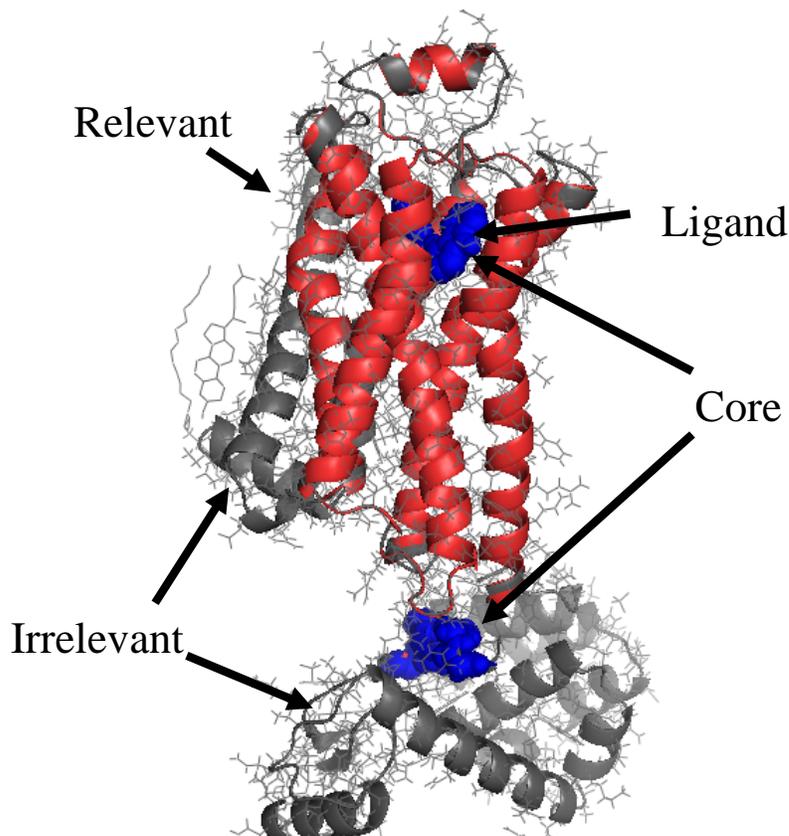


Figure 5.21: Allosteric analysis (relevant regions) on Human beta-2 adrenergic receptor (pdb id: 2rh1). Two sites are shown in blue spheres, which together define the core G_c . The relevant region (i.e. $G_R \setminus G_c$) of the two sites (core) is shown in red, and indicates that two sites are involved in rigidity-based allosteric communication. The relevant region (allosteric communication pathway) passes through most transmembrane helices, except helix 1, which points out these helices to be important for transmitting communication. Relevant region also include part of the extracellular loop regions (top) of the protein, which are also known to be important binding sites in this protein [26]. The irrelevant regions are shown in gray and not important for the communication of the two sites.

²⁸From top view of the protein, by visual inspection using Pymol, the binding site somewhat resembles a barrel like cavity generated by the bundle of helices.

Recall that any changes (i.e. breaking/forming bonds) in the relevant region (i.e. rigidity-based communication pathways) will modify and/or disrupt the communication between the two sites, as the relevant region is constraining the motions (DOF) of the two combined sites (core). We observe that the relevant region includes part of the intracellular and extracellular loop region of the protein²⁹. The extracellular loop regions (which for beta-2 adrenergic protein includes small helical segment, see top of Figure 5.21) of GPCRs are generally known to be important binding sites, which are responsible for the recognition of many ligands, and often bulkier ligands that cannot penetrate deep to the binding site cavity will bind at these loop regions [21, 26]. In fact, it has been suggested that the entire extracellular surface of a GPCR can in general be considered a potential binding site for biologically active molecules, both other proteins and small molecules such as drugs [26].

As changes in the relevant region can modify the rigidity based signal transmission to the intracellular region, our finding of the extracellular loops as part of the relevant region is significant, as these regions can serve as potential allosteric sites. We can envision important applications of this, as small-molecule compound (i.e. drug) could be engineered to inhibit a receptor by interfering with the relevant regions, rather than directly targeting the ligand binding site.

Since the relevant region (communication pathway) of the two sites includes most transmembrane helices, we suspect that helices which are part of the relevant region should have a tightly constrained connection between them (i.e. a restricted number of DOF). To investigate this further, we determined the pairwise relative DOF between all 6 rigid clusters (helices). In other words, we checked for coordinated motions between pairs of helices. Recall that two rigid clusters are in a coordinated

²⁹Note if instead of the ligand we pick a site in the extracellular loops, where the second site is the same site in the intracellular region, we still find direct communication (i.e. relevant region connecting the two sites), further supporting that these extracellular loops are communicating with the intracellular regions and can serve as potential allosteric binding sites.

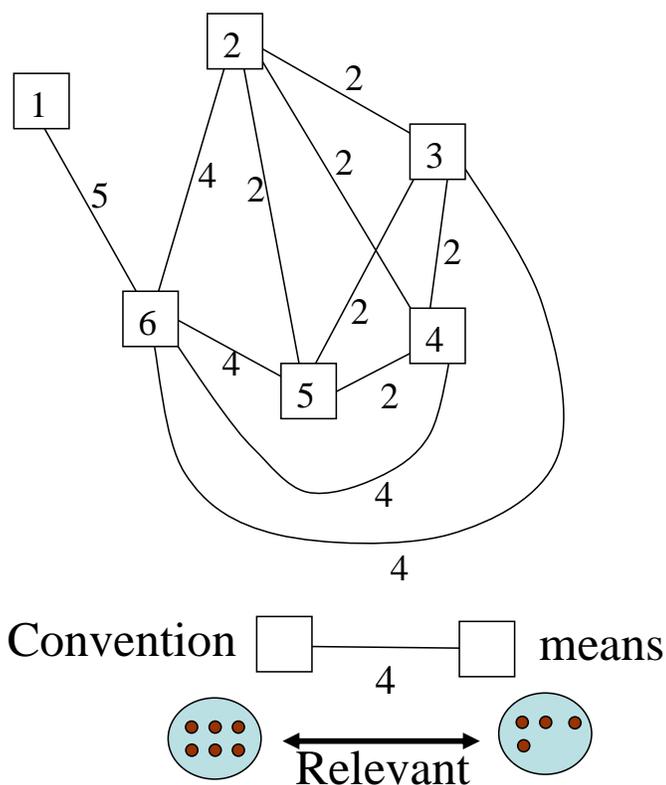


Figure 5.22: Coordination graph of (rigid) helices for Human beta-2 adrenergic receptor (pdb id: 2rh1), where each block (vertex) corresponds to a rigid helix with the same numbering used as in rigid cluster decomposition in Figure 5.20. When two rigid clusters (helices) are in a coordinated motion, we connect the two corresponding vertices with an edge, and assign a number (weight) to the edge which is the relative DOF between the two rigid clusters. When this number is 6 (i.e. two rigid clusters are not in a coordinated motion) we do not connect the corresponding vertices with an edge. Below the graph, we have shown in more detail the meaning of this convention. For instance, two rigid clusters 5 and 6 are connected by an edge with value 4. More specifically, in the output of the pebble game, we could draw maximum 10 pebbles to its ends, as is for instance shown here. As two rigid clusters are in a coordinated motion, there will also be a relevant region between them. In this protein we see that there is a significant number of pairs of helices which are in a coordinated (i.e. restricted) motion.

motion, if in the pebble game output the maximum number of pebbles we can draw back to a rigid cluster is less than 6, while 6 free pebbles remain frozen at the other

cluster. That is when one cluster is frozen, the other one has less than 6 DOF (trivial rigid body motions).

In Figure 5.22 we visually display the output of pairs of rigid clusters (corresponding to helices) which are in a coordinated motion, which we call a *coordination graph*. The numbers corresponding to the vertices, correspond to the same numbering of rigid clusters (helices) given in Figure 5.20. When two (rigid) helices are in a coordinated motion, in the coordination graph, we connect the two vertices (which represent the two rigid clusters, i.e. helix or two helices if they are both part of the same rigid cluster) by an edge (indicating coordination) and assign to each edge the relative DOF count between the two rigid clusters. As we are assigning numbers (weights) to each edge, the coordination graph can be viewed as a weighted graph. If two helices are not in a coordinated motion (i.e. both can simultaneously get 6 free pebbles), to avoid clutter, we do not connect their corresponding vertices, as every one of these edges would get assigned value 6.

Looking over the coordination graph for beta-2 adrenergic receptor, we find that 5 out of the 6 rigid clusters (helices) are all in mutual coordinated motions, while helix 1, as expected, is excluded and is only in coordinated motion with helix 6. Having coordination between pairs of helices means that their relative DOF (motions) between them are tightly constrained, and having a tight (constrained) connection between helices is an important property for communication between the two sites (see below). We will continue this discussion below and make further comments on coordination graphs.

We now apply our rigidity-based allosteric techniques to a GPCR A_{2A} adenosine receptor (also known as the ‘caffeine receptor’ as caffeine inhibits its action), that has just recently been solved in both activated and inactive states (i.e. agonist

and antagonist-bound, respectively) [113, 215]³⁰. This will give us an opportunity to compare the rigidity/flexibility of the two structure. More importantly, we can perform allostery analysis and coordination of helices in both the active and inactive states. Any significant differences that are found will be due to the effects of the type of a ligand that is bound (i.e. agonist-bound or antagonist-bound) to the receptor.

Remark. We chose to test this protein, as this is the only structure of a GPCR that was solved in both active and inactive states, with three structures solved in the active state. It would gives us the most information as we can compare our predictions on both active and inactive structures. The authors of [113, 215] explicitly mention that in the active state the three structures are solved with agonist ligands, and in the inactive state with the antagonist ligand. No other protein was found to have this clear distinction. So, the selection of a protein was not an arbitrary choice. ■

In Figure 5.23 we show the output of FIRST on A_{2A} human adenosine receptor in the active state (pdb id: 3qak) [215]. First predicts that all 7 transmembrane helices are separate rigid clusters. We should note that helix 7 is not entirely rigid, as it is composed of two rigid clusters, where the rigid cluster closer to the extracellular region is significantly larger. The (agonist) ligand is shown in orange sticks. Notice how the ligand extends deep into the binding pocket where it interacts with several surrounding helices.

In Figure 5.24 we have displayed the output of the relevant region analysis on this protein, where again one site is the ligand and another site is located at the intracellular region. The two sites are shown as blue spheres. The relevant region of these two combined sites (core) is shown in red. The two distant sites in this protein are predicted to be communicating with each other as they are connected by

³⁰These receptors are involved in controlling inflammatory responses, they regulate the effects of neurotransmitters in the brain, cardiovascular and immune systems [88, 113, 215]. They are often a drug target for asthma and chronic obstructive pulmonary disease and are of particular interest as a drug target for Parkinsons disease [215].

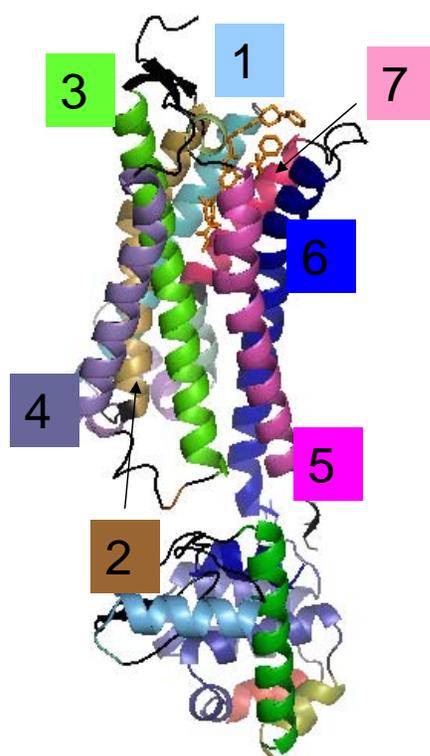


Figure 5.23: FIRST rigid cluster decomposition of A_{2A} human adenosine receptor in the active (agonist-bound) state (pdb id: 3qak). All transmembrane α -helices are decomposed into separate rigid clusters. The ligand (located at the extracellular end of the protein) between the bundle of helices is shown in orange spheres, which penetrates deep into the binding cavity where it creates important interactions with the helices.

a relevant region. Interestingly, the relevant region involves most helices, and as in beta-2 adrenergic receptor the N-terminal helix 1 is irrelevant so it is not part of the rigidity-based allosteric communication pathway. All the remaining helices (except a C-terminal part of helix 7) are involved in communication transmission between the two sites. The communication pathway (relevant region) between the two sites is 2-connected, so this protein likely has allostery type 2 communication as well (see end of this section for more detailed discussion).

To compare these predictions with the inactive state of A_{2A} human adenosine receptor, we have performed the similar analysis on the inactive structure. The

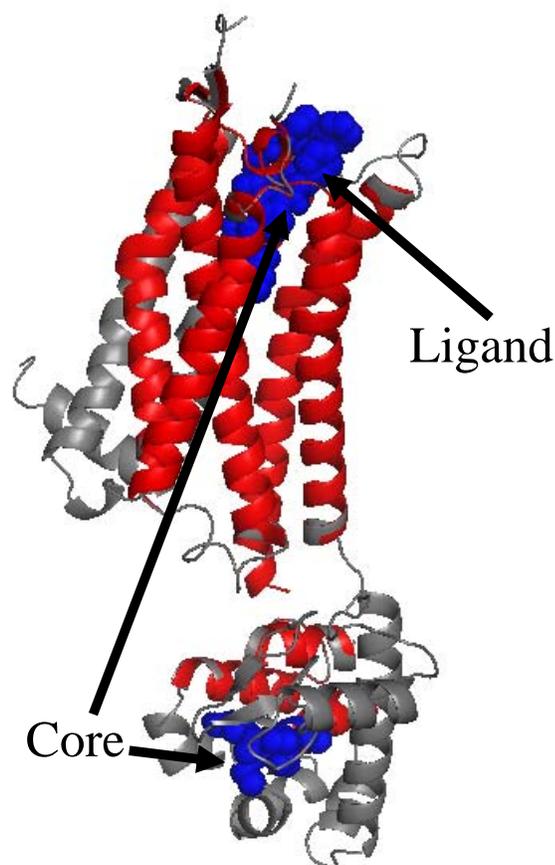


Figure 5.24: Allosterity analysis (relevant regions) on A_{2A} human adenosine receptor in the active state (pdb id: 3qak). Two sites are shown in blue spheres, which together define the core G_c . The relevant region of the two sites (core) is shown in red, and indicates that two sites are involved in rigidity-based allosteric communication. The relevant region (allosteric communication pathway), which is constraining the relative DOF (motions) of the two sites, passes through most transmembrane helices. The irrelevant regions are shown in gray and are not important for this communication.

FIRST output on the inactive form (pdb id: 3eml) of this receptor is shown in Figure 5.25. Note how the (antagonist) ligand is significantly smaller and does not extend as deep into the ligand-binding pocket. As with the active state, FIRST finds a similar rigid cluster decomposition, with all seven helices forming separate rigid clusters, so no significant differences are observed. As was done in the active state, we pick the ligand (in this case an antagonist) as one site and the second site in the intracellular

region, which defines the core. When we apply the relevant region analysis, we now find there is no relevant region between the two sites. In fact, no matter what we pick to be the site in the intracellular region, we find no relevant region connects it with the ligand or its binding pocket.

In summary, with our rigidity-based allostery predictions, we found that in the active form of the protein, the two distant sites are involved in rigidity-based allosteric communication. In contrast, in the inactive form there is no rigidity-based communication (see below for further discussion and other likely reasons).

To obtain a deeper understanding of these results, we have computed the coordination graphs for both the active and inactive states of A_{2A} human adenosine receptor. Since in the active state the two sites are communicating with each other, as they are connected by a relevant region, which also encompasses many helices, the active state should be generally more constrained (rigid) than the inactive state. We anticipate that helices in the active state are involved in coordinated movements. To check this, in Figure 5.26 we have displayed the coordination graph for the active state protein. We observe that significant number of helices are in a coordinated motion (i.e. freeze one, the motions on the other one are restricted (has less than 6 DOF)) where many pairs have 3 relative DOF between them.

In contrast, in the inactive protein (pdb: 3eml), the results are drastically different, as we found that no pairs of helices are in a coordinated motion (i.e. each pair of helices would get assigned value 6 in the coordination graph).

To recap, in the active state of A_{2A} human adenosine receptor we have found the two distant sites (extracellular and intracellular) are in a rigidity-based allosteric communication as they are connected by a relevant region, whereas in the inactive state we found no communication. As anticipated, in the active state we found that the helices are much more tightly constrained with respect to one another. So, ligand

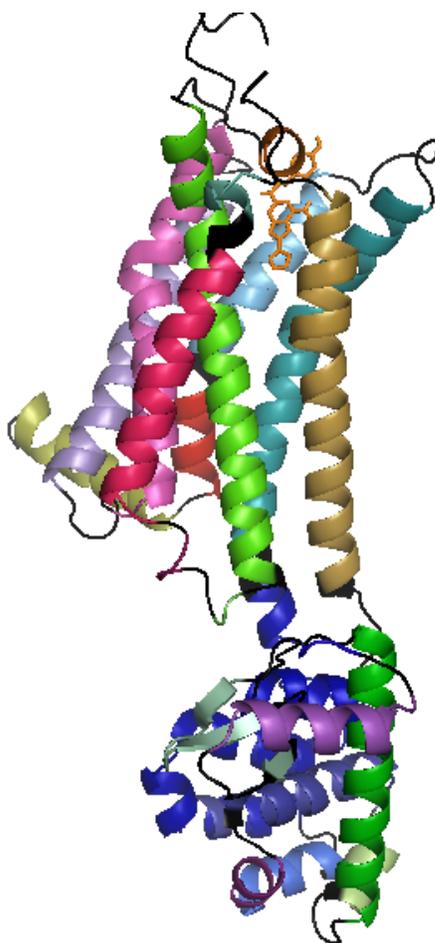


Figure 5.25: FIRST rigid cluster decomposition of A_{2A} human adenosine receptor in the inactive state (pdb id: 3eml). The ligand (located at the extracellular end of the protein) between the bundle of helices is shown in orange spheres. Note how this (antagonist) ligand does not penetrates very deep into the binding cavity.

(agonist) in the active state causes a relative constraining (tightening) of the helices, as significant number of pairs of helices are in a coordinated motion (i.e. have a few relative DOF) ³¹. On the other hand, in the inactive state there is no coordination among helices, which means that the (antagonist) ligand in the inactive state leaves helices with a much more looser connection (i.e. freeze any helix, then the other

³¹Coordination of helices occurs over a relevant region, and we know that relevant regions are highly constrained (recall that rigidification of the core, will also rigidify the entire relevant region together with the core into a single rigid cluster), and coordination of helices is suggesting a sharing of a limited number of DOF.

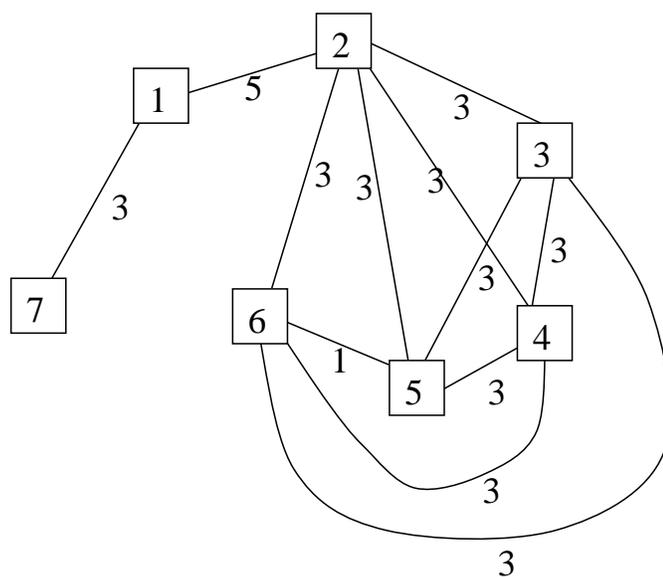


Figure 5.26: Coordination graph for A_{2A} human adenosine receptor in the active state (pdb id: 3qak) (see above for explanation of the coordination graphs). In this protein we see that a significant number of helices are in coordination (i.e. coordinated motions).

ones still have full 6 DOF, any helix is freely moving relative to the other helix). We can clearly see that coordination graphs are very useful as they can tell us how tight or loose the connection between the helices is in the presence of different ligands, and also the potential for rigidity-based allosteric communication. When there is no coordination it would indicate that communication is very unlikely (see below for more commentary).

These findings of the active state having a more tight connection between the helices, agree with a recent study which has found that active states of many allosteric proteins are generally more rigid than the inactive states [146].

Our findings of coordinated motions (i.e. restricted motions, tightening) among helices in the active structure of A_{2A} human adenosine receptor, and not in the inactive structure is supported by observations of authors that have experimentally determined

these structures [113, 215]. We now summarize some of these experimental findings, and immediately draw further comparisons and evidence for our predictions:

- (1.) As we had visually observed, and as authors confirm [215], the ligand (i.e. agonist) in the active state (which is significantly larger than the ligand in the inactive state) penetrates the binding pocket (i.e. between the bundle of helices) much more deeply than a ligand (i.e. antagonist) in the inactive state, and occupies most of the ligand-binding cavity. The important consequence of this is that the ligand in the active state forms several additional hydrogen bonding interactions with helical residues (residues 88, 250, 277 and 278), whereas the antagonist ligand in the inactive state of this protein does not interact with all these residues. Authors report that these additional interactions are important for activation of this protein [215].
- (2.) Mutation studies show that mutations of the residues mentioned above disrupts the ligand binding, indicating that they play an important role in the ligand binding [104, 215].
- (3.) These additional interactions (hydrogen bonds) induced by the larger (agonist) ligand in the active state, cause a contraction of the binding pocket and an increased relative tightening between the helices [113, 215].
- (4.) The presence of these additional interactions cause important (yet relatively small) shape changes at the binding pocket. These small shape changes are presumed to promote shifts and ‘coordinated movements’ of the transmembrane helices (some which resemble a piston-like motions, in addition to other motions) which in effect cause important shape changes (i.e. allosteric) at binding regions at the intracellular parts of the protein [215]. This is believed to be a common feature in activation of all GPCRs [113].

Returning to the discussion of our computational prediction of allostery on adenosine receptor, FIRST correctly identifies these additional hydrogen bonds between the activated receptor and the agonist ligand that were reported by authors, so they were captured in the rigidity analysis. More importantly, in our allostery analysis, all these hydrogen bonds are declared as relevant (i.e. form relevant bridges between the corresponding helices and the ligand in the relevant region of the two sites). This indicates that these hydrogen bonds are important for the rigidity-based allosteric communication and that they are also constraining the motions of the helices, resulting in restricted, coordinated motions of helices.

In summary, our analysis of this protein, demonstrates that we can predict communication between two distant sites in the active state, and lack of communication in the inactive state. Furthermore, we have predicted that coordinated motions of helices is also an important feature for communication of the two sites. Being able to predict detailed information about regions (i.e. relevant residues and hydrogen bonds) in the protein that are important for this allosteric communication, can provide guidance for mutation studies, and our predictions are experimentally verified for several relevant residues and relevant hydrogens bonds listed above. We predict that changes (i.e. mutations, breaking of hydrogen bonds) in the other parts of the relevant region of this protein can also impact the allosteric communication, but this would have to be experimentally verified.

To further strengthen our understanding of how ligand binding can modify coordination among a collection of helices, let us consider a simple toy model in Figure 5.27. In this example, let us assume that each green object represents a rigid body, such as a rigid helix in transmembrane receptors. For sake of simplicity, in this example we chose to have 4 such rigid bodies (helices). We have placed some vertices (atoms) and edges (bars) between the hypothetical helices which represent

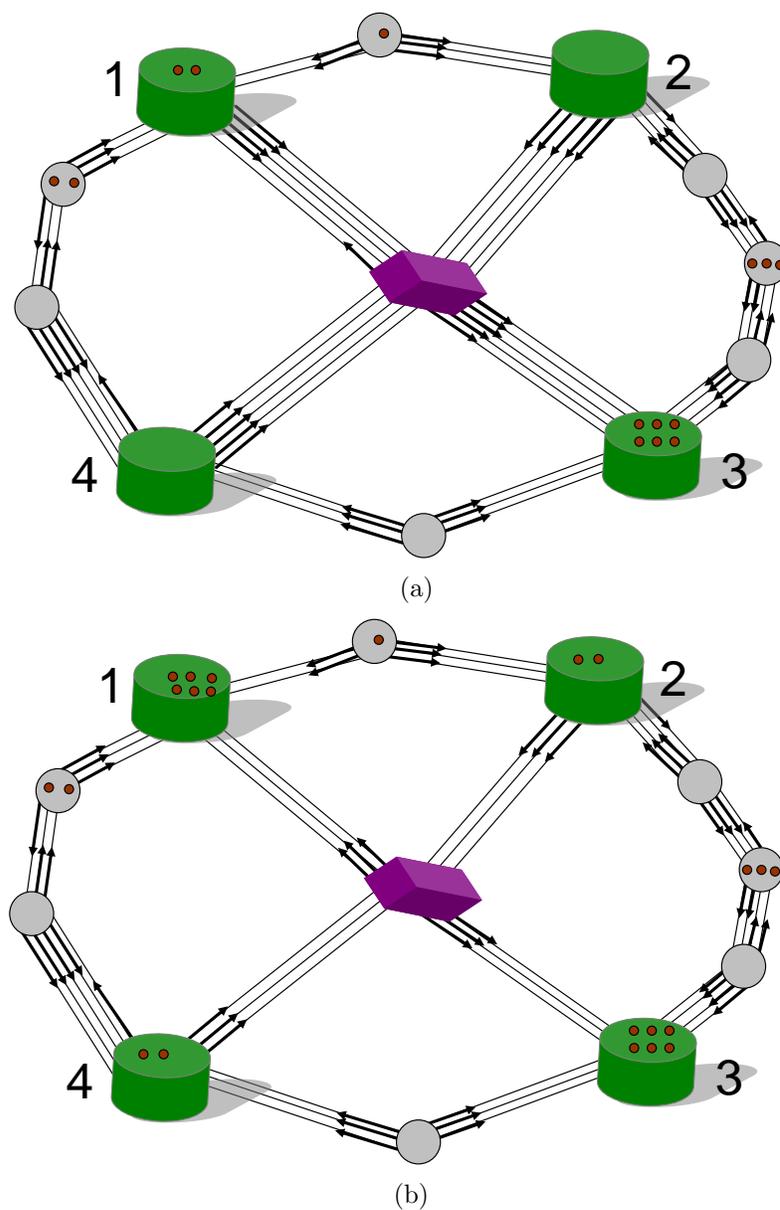


Figure 5.27: Schematic representation of how coordination can change between four hypothetical helices (shown in green) when ligand (purple) forms different number of constraint with the helices. All bodies (helices, atoms (grey) and the ligand) are assumed to be rigid. Treating each rigid body as a vertex, and bars (constraints) between them as edges, the output of the $6|V| - 6$ pebble game algorithm is shown. In (a) each pairs of helices is in a coordinated motion, with 2 (internal) relative DOF. In (b) the ligand forms less constraints (a looser connection) with the helices (3 bars vs 5 bars) and as a result no two hypothetical helices are in a coordinated motion. In this example the looser ligand caused the helices to change from coordination to no coordination.

the interactions and connections between the helices. The purple object represents a ligand, and for simplicity let us assume that it is rigid as well. Treating each rigid body as a vertex, and bars (constraints) between them as edges, we have played the $6|V| - 6$ pebble game algorithm on the multigraph. Note that we have sometimes used less than 5 bars between the pair of rigid bodies ³².

With the presence of ligand in Figure 5.27 (a), every pair of green bodies (hypothetical helices) is in a coordinated motion with two relative DOF between them (i.e. freeze one green body, any other green body has 2 remaining DOF). So the ligand has created a tight connection within the helices ³³.

Now, if we the ligand forms less constraints with the helices, providing a looser connection between the hypothetical helices (Figure 5.27 (b)), there is no more coordination (each pair of green bodies can simultaneously get 6 free pebbles i.e. freeze any green body, the other once have full 6 DOF). Note, if we remove the ligand completely, then freezing any three green bodies simultaneously will still leave 6 free pebbles (DOF) at the other green body. So, we can see from this example how having a tighter (more constrained) connection with the ligand, can shift the helices from not coordinated to coordinated (i.e. coordinated motions). It is also possible that a looser-bound ligand can still keep the helices in a coordinated motion but change the amount of coordination. For instance, in this hypothetical example, if the ligand

³²Having less than 5 bars between two rigid bodies is completely reasonable, and we can equivalently replace less than 5 bars with appropriate length perfect bridges. For instance, one bar between the pair of rigid bodies is a perfect bridge of length 5, 2 bars is a bridge of length 4, etc. (see previous chapter on the discussion on bridges).

³³In this hypothetical example, we have placed the purple body (ligand) between all green bodies (helices), restricted their relative motions. Attaching a ligand to just a single helix is not consistent with the actual observations in GPCRs as typically ligands interact with two or more helices [26, 215]. In addition, in [34] it is hypothesized that in allosteric units the ligand should bind between at least two rigid bodies, and that motions of the two rigid bodies should perturb the binding site. This also makes sense in terms of the rigidity analysis, as attaching a ligand to a single helix (rigid body) would not alter the rigidity.

(purple) is connected to each helix (green object) with 4 edges instead of 5, the relative DOF count between any pair of hypothetical helices will go up from 2 DOF to 4 DOF.

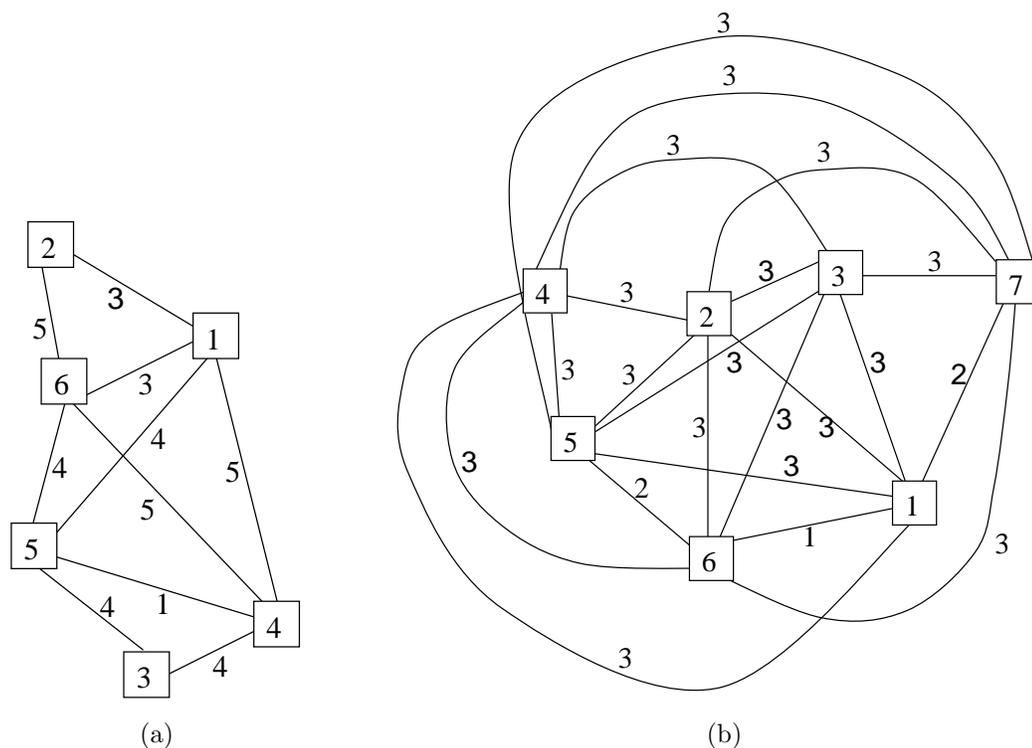


Figure 5.28: Coordination graphs for A_{2A} human adenosine receptor in two active states (pdb id: 2ydo) (a), and (pdb id: 2ydv) (b) (see above for explanation of the coordination graphs). In both of active structures there is significant coordination (i.e. pairs of (rigid) helices are in a coordinated motion). In the second structure, all pairs of helices are in a coordinated motion, indicating very high coordination. That is freeze any helix, all the other helices will have a restricted motions (i.e. less than 6 DOF). The first 5 helices (rigid clusters) are almost identical (i.e. same residue numbers) between the two structures.

We shall now further investigate and predict the importance of coordination among helices in the active structure of the protein we have been discussing. We have shown the coordination graphs (Figure 5.28) of the two additional crystal structures (i.e. with different (agonist) ligands) that were very recently solved in the active state of A_{2A} human adenosine receptor (pdb ids: 2ydo, 2ydv) [113]. Looking over

the coordination graphs for these two activated states, we find that there is again significant coordination among the pairs of helices. This finding provides additional evidence that in the active states of this GPCR protein there is coordination (tightening) among the pairs of helices. It is interesting to observe that in structure (b) with 2ydv all pairs of helices (rigid clusters) are in a coordinated motion. As was the case in the previous active structure (3qak) similar residues in both of these structure are also interacting with their corresponding (agonist) ligands, forming hydrogen bonds, which were detected by FIRST and also come up as relevant, meaning that they are restricting the number of relative DOF (motions) between the helices.

As we had found with the active structures for previous GPCRs, both of these active structures can facilitate communication (i.e. existence of a relevant region) between two widely separated sites via the transmembrane helices. Moreover, the fact that in all three active structures of this protein, there is a tight connection between the pairs of helices (i.e. coordinated motion) suggests that coordination is likely an important property for facilitating allosteric communication. Drawing from these findings and similar patterns that we have observed on some other tested GPCRs (i.e. existence of coordination among helices), we speculate that having some coordination (tightening) between a pair of helices is a necessary condition for communication (i.e. changes in rigidity, shape change, coordination) between the extracellular and intracellular binding sites. If there is no coordination between the helices (i.e. very loose connection between the helices, as was the case for the inactive state of adenosine receptor), it would be very unlikely for the two sites to be in communication. It is then not surprising that in [215] authors report that ligand (agonist) induced ‘coordinated movements of helices’ in the active state of adenosine receptor cause shape changes at the intracellular binding regions which are important for receptor activation.

Once more GPCRs structures become available in both active and inactive states, and assuming that our speculations of coordination of helices and general rigidity based communication via relevant regions is indeed a preserved property in allosteric communication in this class of proteins, we can foresee further important applications of these methods.

Assume we are given a single GPCR structure ³⁴ with a bound-ligand (at the extracellular binding pocket) whose effect on the receptor is not know (i.e. does it activate or inactivate the receptor?). One possible question is if there is an allosteric communication (with that bound ligand) between the two distant binding sites. In such case, we would first obtain coordination graphs of the pairs of helices, and if in the coordination graph we find no pairs of helices are in a coordinated motion with each other, then we would predict that it is highly unlikely that there is allosteric communication between the intracellular and extracellular binding regions with that particular bound ligand. We then predict that with that bound ligand the receptor is in the inactive state. If on the other hand, there is strong coordination among the helices, as was for instance the case with all three active states of adenosine receptor, we then expect to find the relevant region between the two binding sites, and with the presence of the relevant region (rigidity-based allosteric communication) we would predict that allosteric communication is possible, and that the relevant regions are the predicted allosteric communication pathways. Furthermore, if we find communication between the two sites, we can hypothesize that the protein (with that bound-ligand) is in the active state.

Thus, we propose that our techniques could be an initial filter for potential allosteric communication, where it would then be appropriate to do further experimental investigations. We are speculating based on the patterns and observations

³⁴Recall that with our tools using the pebble game algorithm and relevant region analysis only a single protein structure is needed to test for communication.

we have made on the few GPCRs that we have tested, with the available supporting experimental evidence. Certainly, this would have to be further investigated with more GPCRs (as more become available) as we have done with some structures in this chapter.

In summary, we have demonstrated how rigidity-based allosteric communication can be transmitted across the membrane through the bundle of transmembrane helices in two different GPCRs, and also that relative tightening of helices (i.e. existence of a coordinated motions) is likely a necessary property for allosteric communication.

Allostery type 2 and type 3 communication

So far we were mainly testing for allostery type 1 communication by checking for a relevant region between the two sites. Allostery type 1 communication is very important as it is the necessary condition for type 2 and type 3 communication. In other words, the lack of type 1 communication between the two sites can be used as an initial check to conclude that type 2 and type 3 communication are not possible.

Given that the active state of adenosine receptor has type 1 communication suggests that type 2 and type 3 communication are possible. Moreover, since the communication pathway (relevant region) between the two sites is at least 2-connected, the active state likely has allostery type 2 and type 3 communication. In contrast, in the inactive state there is no type 1 communication, so type 2 or type 3 communication are not possible.

In figure 5.29 we have displayed the predictions for type 2 and type 3 communication on the active state (i.e. with agonist bound ligand - adenosine ³⁵). We have found that the active state has type 2 and remarkably type 3 communication.

³⁵Adenosine is an endogenous agonist as it is a naturally produced compound in the body which activates the adenosine receptor.

For type 2 communication we have selected the side chain atoms that are interacting (i.e. with hydrogen bonds) with the ligand as one site (as reported in [113, 215]) and another site as the side chain atoms between two helices at the intracellular regions and then applied the Algorithm 5.5.3. No atoms in the two sites are part of the rigid α -helices (i.e. sites are flexible binding pockets). We found that $\text{DOF}_{\overrightarrow{AB}}$ is positive (i.e. rigidification of one site causes a decrease in DOF at the other site), confirming there is type 2 communication. In Figure 5.30 we have provided the dilution plot and indication of the hydrogen bond energy cutoffs where transmission of DOF occurs. Transmission begins at a cutoff of -0.944 kcal/mol, with 3 DOF being transmitted (i.e. $\text{DOF}_{\overrightarrow{AB}} = 3$) upon rigidification of one site. At this cutoff a total of 253 hydrogen bonds are included as constraints. The transmission continues up to -1.387 kcal/mol cutoff where 223 hydrogen bonds remain as constraints. So, transmission of DOF is stable over a substantial range of energy cutoffs. In this range the maximum amount of possible DOF that can be transmitted increases from the initial 3 DOF beginning with a cutoff of -0.944 kcal/mol to 4 DOF and then once more hydrogen bonds are excluded it gradually decreases back to 3, 2 and 1 DOF transmission.

For type 3 communication, only the intracellular site (same intracellular site as in type 2 test) was declared as the core and the Algorithm 5.5.2 was applied. The relevant region of the intracellular site reaches the adenosine binding site and the adenosine ligand. So, removal of adenosine or breaking the hydrogen bonds that are formed with the ligand increases the free pebble (DOF) count at the intracellular site. Alternatively, binding of adenosine restricts the possible motions (DOF) at the intracellular site.

To confirm our earlier anticipation, we found no type 2 or type 3 communication in the inactive state of adenosine receptor (pdb id: 3eml).

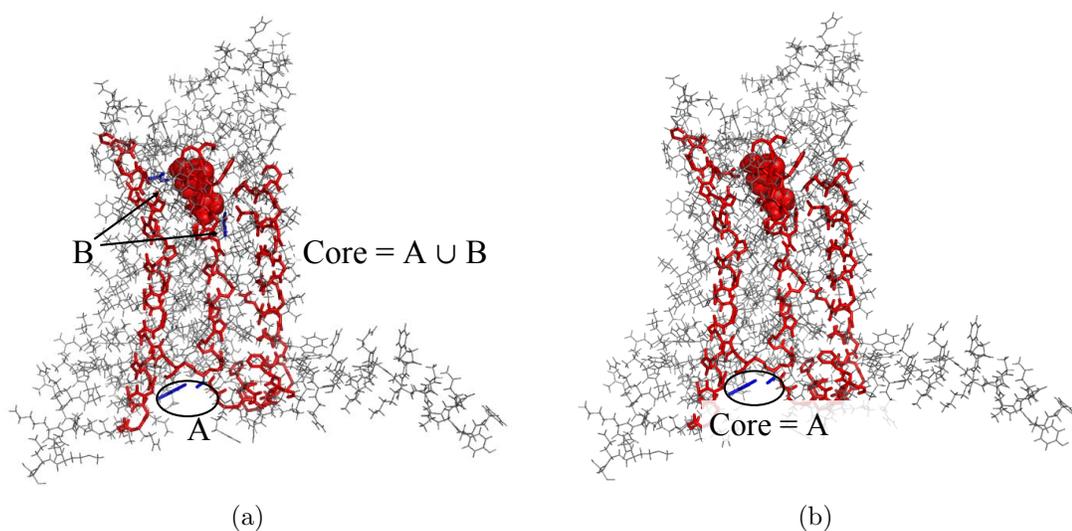


Figure 5.29: Allostery type 2 (a) and type 3 (b) communication in GPCR of an active state adenosine receptor (pdb id: 2ydo); energy cutoff -1 kcal/mol. The core is shown in blue and the relevant region (communication pathway) in red, irrelevant regions in gray and the ligand is depicted with spheres. (a) Rigidification of one site causes 3 DOF to be removed from the other site (i.e. $\text{DOF}_{AB} = 3$). (b) Relevant region of site A includes site B (and the ligand), confirming type 3 communication.

In conclusion, the presence of types 1, 2 and 3 rigidity-based communication in the active state of adenosine receptor and lack of any communication in the inactive state, suggests that rigidity-based communication is a plausible mechanism of activation of GPCRs.

5.7 Concluding remarks and future work

The search for mechanisms that are used to transmit shape changes in allosteric proteins has been an area of active study for some time [68, 190], so far such mechanisms eludes understanding, and this remains one of the most important open problems in protein science.

In this chapter we have introduced several combinatorial rigidity-based allostery models and detailed corresponding algorithms that can detect transmission of

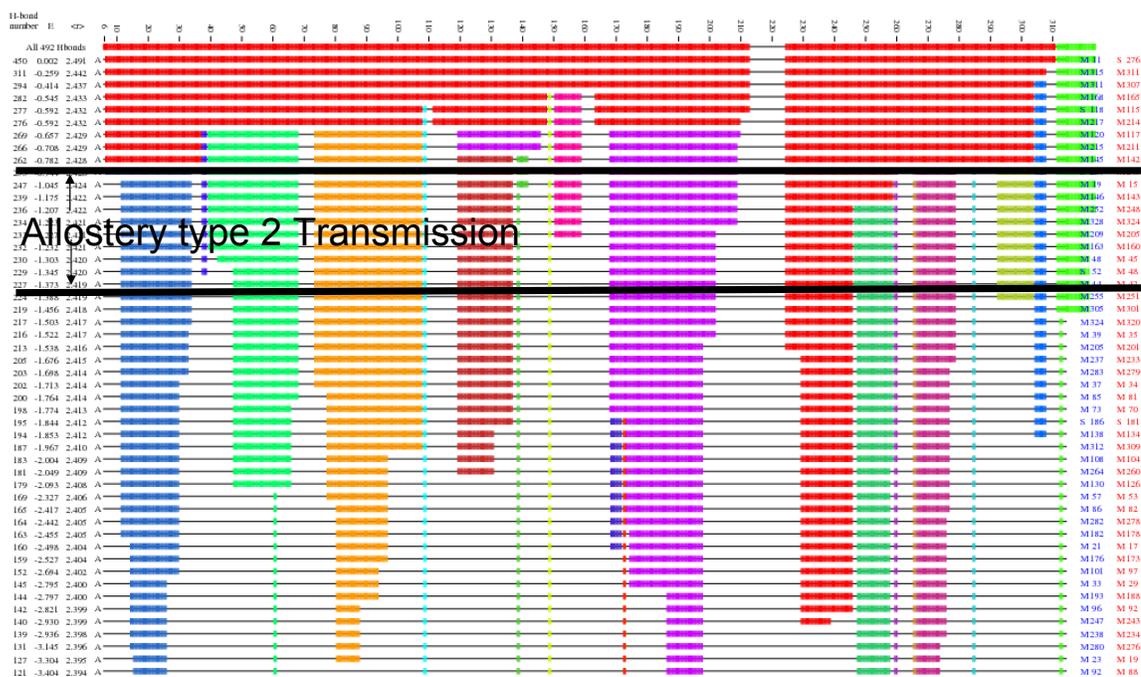


Figure 5.30: Dilution plot of the active state adenosine receptor (pdb id: 2ydo). Allosteric type 2 communication between sites *A* and *B* (Figure 5.29 (a)) begins at energy cutoff -0.944 kcal/mol (253 hydrogen bonds - top thick line) and transmission continues up to -1.387 kcal/mol (223 hydrogen bonds - bottom thick line). The amount of transmission varies between 1 and 4 DOF.

rigidity (change in shape) between distant sites, and other types of rigidity-based communications. The algorithms we have developed utilize the pebble game algorithm and various extensions of the relevant region detection algorithm. We have initially illustrated these models and algorithms on several small toy models and graphs. A significant number of examples were provided to demonstrate various scenarios that can occur in rigidity-based allosteric communication. Some toy models were used to illustrate how small shape changes can lead to (sometimes large) shape changes at a distant site. We have seen examples which show that rigidifying one site may completely rigidify the other site or leave residual DOF. Other examples illustrated that in some cases a certain number of constraints (edges) may need to be added first at a site in order to begin (transmission) to then reduce DOF at the second site. The

examples and toy models provide us with an increased understanding of the type of behaviour we could expect in rigidity communication at a distance.

In our rigidity-based allostery predictions, not only are we able to detect if there is communication between two distant sites, the algorithms are also designed to predict which regions are important for this communication (i.e. allosteric communication pathways), which essentially correspond to relevant region predictions.

We have seen that the relevant region detection is an extremely useful tool for finding out the regions of the protein that are used to transmit rigidity information (i.e. changes in rigidity and flexibility, coordinated motions) across graphs and proteins. Detection of a relevant region for two sites can have important consequences in mutation studies, as change (mutations) in the relevant regions should affect the communication of the two sites. The relevant regions can potentially also be used to detect new allosteric sites or new engineering of allostery into proteins that were thought to not be allosteric, as any site that we find is communicating (via relevant regions) with the active site could be a potential allosteric site. In a more general setting, we hypothesize that the relevant regions can be used to predict the long range effects of mutations on the rigidity, flexibility and allostery.

We have verified various mathematical properties of the rigidity-based allosteric communication pathways. One result that stands out is that 2-connectivity in the allosteric region is a necessary condition for allostery type 2 communication (i.e. reduction of flexibility at a distance - shape change) and is conjectured for allostery type 3.

We have demonstrated rigidity-based allosteric communication between distant sites on some actual protein structures, suggesting that rigidity-based communication could be one possible allosteric mechanism. The two examples we have looked at were calmodulin and several different structures of G-coupled protein receptors (GPCRs).

Our algorithms predict that the active state of adenosine receptor has all three types of rigidity-based communication.

We can envision a wide range of further possible directions that should be investigated in future research.

Most of our algorithms were designed with the assumption that there are only two sites that can potentially communicate with each other. However, in many larger proteins, especially oligomeric proteins, many more sites could be involved in allosteric communication [190]. It is conceivable that a change in rigidity at one site can alter the rigidity at several other sites. We could certainly use our algorithms to study the pair-wise communication between all possible sites, and in some cases (i.e. when individual sites are rigid) we could simultaneously handle more than two sites. As part of future work, it would be helpful to systematically extend our current methods and algorithms in order to simultaneously monitor rigidity allosteric communication of several sites. It would be also interesting to look for any patterns and general characterizations of the (potentially intersecting) communication pathways involving a larger number of sites.

We anticipate that our rigidity models and algorithms can offer new predictions and insights into many other proteins, but this would need to be investigated in future work. Using the database of a collection of allosteric proteins solved in both active and inactive states [33], we have started initial investigations using the approaches given in this chapter, using a more general bioinformatics approach. Due to the nature of the pebble game algorithm, we can give rapid predictions of rigidity-based communication, which makes it suitable for high-throughput analysis. In short, we aim to consider all proteins from this database and consider rigidity-based allosteric interactions between all possible pairs of sites. All possible pairs of sites will be considered since many proteins are oligomers which have multiple binding sites (both

allosteric and active) that could be interacting with each other. A site is defined as residue(s)/ligands within a certain distance of a bound-ligand that is given in the structure or residues that are within a certain cutoff distance of the catalytic sites. To facilitate fast analysis, the residues of the catalytic (active) sites will be taken from the Catalytic site atlas [141] database. We suspect that insights from this study will further confirm the importance of rigidity analysis and rigidity-based communication as one likely mechanism in protein allostery.

The number of solved GPCRs is still very small, but more are appearing over the last few years. We have shown some encouraging results that illustrate rigidity-based communication in this important class of proteins. As structure determination of membrane proteins powers ahead and as more structures of GPCRs are determined, particularly in both active and inactive states, one of the important next steps in the future work, will be to continue to apply the procedures that were developed in this chapter. It would also be interesting to see if coordination (i.e. coordinated motions) of helices is a conserved property for allosteric communication in GPCRs.

One of the natural next steps in this work is to move from the pure combinatorial analysis (i.e. pebble game) and snapshot (infinitesimal) predictions (that are inherent in FIRST analysis) and incorporate the more complicated geometric properties (for instance collisions) and perform simulations of motions with programs such as FRODA and coarse graining MD simulations methods. Our predictions of relevant regions (rigidity-based allosteric communication pathways) of two sites can be valuable when trying to simulate the allosteric motions of two sites. The focus of the simulations should be just on the two sites and their relevant region, where the irrelevant regions could be ignored. Limiting the simulations to significantly reduced, yet biologically relevant degrees of freedom, should improve the computational efficiency of simulations. In this way, the simulations and computer resources could be

focused only on simulating the relevant regions that are responsible for constraining the motions of the two potentially communicating sites.

We had briefly introduced the piston-type motion hypothesis of transmembrane helical receptors. Our work did not attempt to address this hypothesis, as all our tools rely on the combinatorial predictions of rigidity and flexibility. To further probe this hypothesis would require investigations of the geometry (i.e. how do tightly packed rigid cylinders ³⁶ move with only a few DOF) and simulations of actual motions [208]. Since we have observed that many helices are in a coordinated motion (i.e. highly constrained motions) in several active state GPCRs, this coordination could become very useful as simulation of these coordination motions should be significantly less computationally intensive as they have a smaller number of DOF. So, being able to predict which structures have coordinated motions among the pairs of helices, can identify good test cases that can be further investigated using motion simulation methods (i.e. FRODA, coarse grained MD methods) of the type of motions that GPCRs may undergo (i.e. are the motions piston-type sliding motions or combinations of other motions).

It is well known that a number of oligomers with several copies of individual protein chains assemble with symmetry (for instance dimers have simple half-turn symmetry) and that many protein oligomers function allosterically [69]. In addition, previous theoretical results confirm that when a structure has added symmetry this may lead to additional flexibility (i.e. a structure which counts to be generically rigid may become flexible with added symmetry) [157]. Preliminary work has looked at the impact of symmetry on protein rigidity and flexibility, with the message that symmetry related rigidity counts (which capture the hidden extra symmetric motions

³⁶Helices in the membrane proteins, such as GPCRs are known to be very tightly packed [63].

that are missed in regular rigidity counts) and the appropriate symmetry pebble game algorithms should be incorporated into FIRST (see [158, 159] for more details).

An interesting area for future exploration in the applications of rigidity to protein allostery would be to ask the similar questions that we have posed in this chapter, with additional assumed symmetries in the structure. We can ask some general questions:

- (1.) What impact may symmetry have on rigidity (shape) transmission between two (or more) sites located on different symmetric copies of the oligomeric proteins?
- (2.) How would the allosteric communication models and algorithms that we have devised be different/the same, and what changes would need to be incorporated to the symmetry adapted algorithms?
- (3.) How would symmetry alter general relevant regions and what changes may be necessary in the relevant region detection algorithm and other algorithms we have introduced for rigidity-based allostery communication?
- (4.) How may symmetry impact the allosteric pathways of two sites involved in rigidity-based allosteric communication?

Since rigidity-based allosteric communication occurs over relevant regions, which are generally highly constrained with only a few DOF, we expect that any added flexibility (for instance due to symmetry) may impact (change) the communication and even completely disrupt the communication of two sites. We have provided some of the possible speculative questions, and although the general investigations of impact of symmetry on protein rigidity/flexibility are in infancy stages, extending some of the models of rigidity-based allostery communication that we have introduced, by incorporating symmetry, should be another important part of the future work.

In the next chapter we switch themes and turn to predictions of rigidity/flexibility of protein ensembles and we introduce a new computational method for predicting protein hydrogen/deuterium exchange. These are significantly different topics from those in the previous chapters. However, to address these problems we will continue to use similar combinatorial rigidity tools and techniques such as FIRST and the pebble game algorithm, and develop several new extensions and make use of other currently existing techniques.

Chapter 6

Predictions of rigidity and flexibility of Protein ensembles and Hydrogen-Deuterium exchange

6.1 Overview

This chapter is based on the joint work with Derek Wilson and is prepared for journal publication [176].

It is well known by biochemists that proteins can continuously undergo spontaneous sub-molecular unfolding and refolding, or conformational dynamics, even under conditions that strongly favour a well defined native structure [35, 68, 140]. These (local) unfolding events result in a large number of conformers, that differ from each other very slightly [68]. In this context, proteins are better represented as an ensemble of ‘native-like’ structures, and not just as single static (snapshot) structure.

Unlike structures solved by X-ray crystallography, structures solved by Nuclear Magnetic Resonance (NMR) spectroscopy are routinely expressed as an ‘ensemble’

of structures. Molecular dynamics simulations is another source, as they can also generate a collection of conformers. Previous studies using FIRST were designed to analyze the rigidity and flexibility of proteins using a single static (snapshot) structure, whether the structure came from X-ray crystallography or NMR.

In this chapter we introduce a novel FIRST based approach for predicting rigidity/flexibility of the ensemble. In a nutshell, the FIRST-ensemble predictions will be obtained by averaging the hydrogen bonding strengths from the entire structural ensemble (i.e. from models given in the NMR structure) and by refining the mathematical model of hydrogen bonds currently used. Our FIRST-ensemble method will be applied to the NMR structure of protein Acylphosphatase from hyperthermophile *Sulfolobus solfataricus* (Sso AcP). We will validate our FIRST-ensemble predictions with a comparison to independently collected experimental data on site specific hydrogen/deuterium exchange (HDX) experiment (which is an ensemble measurement) of the protein Sso AcP [176, 210].

In the second main contribution in this chapter, we combine our ensemble rigidity predictions with the ensemble solvent accessibility data of the backbone amides and propose a computational method which uses both rigidity and solvent accessibility for predicting HDX. Both solvent accessibility data and FIRST-ensemble rigidity predictions should be valuable tools in probing HDX. However, each has its own advantages and disadvantages (see section 6.3). To get a good prediction of HDX, both analyses need to be carefully combined to give one prediction. To the best of our knowledge, this is a first study that incorporates computational predictions of rigidity and solvent accessibility for predicting HDX.

In summary, we find that our ensemble-based FIRST method, gives a much better prediction of rigidity/flexibility, as confirmed by HDX data, than the traditional single ‘snapshot’ FIRST prediction. Moreover, the computational predictions

of regions that are protected from HDX and those that undergo exchange are in very good agreement with the experimental HDX profile of Sso AcP.

6.2 Introduction

6.2.1 Extending FIRST to Protein ensembles

Protein flexibility can be analyzed accurately and efficiently using FIRST and its main algorithmic component the pebble game algorithm, which detects rigid and flexible region. In numerous studies, it has been thoroughly demonstrated that FIRST gives accurate predictions of flexibility and rigidity in proteins [67, 82, 83, 109]. In the previous two chapters, we have also seen how FIRST, together with our extensions of the pebble game algorithm, can be used to predict hinge locations in hinge-bending proteins, and allosteric interactions. We have given more detailed discussion on FIRST and its methodology in the first chapter.

Protein structures can be viewed as ensembles

One of the limitations of the current FIRST is that it is primarily designed to perform the flexibility analysis on a single structure (snapshot) of a protein. However, protein molecules are certainly not frozen objects with a single conformation. It is now widely accepted that even under native conditions, proteins can undergo local conformational fluctuations, sampling a distribution of native like conformational substates (conformers) [35, 68, 140], referred to as the ‘native-state ensemble’ [68].

Hydrogen/deuterium exchange experiments have illustrated that proteins are best represented as a large collection of slightly different conformational states [154]. The authors of this study argue, if the protein could only adopt a single native structure (i.e. single conformer), any amide that is protected from exchange (i.e. for

instance amide proton is completely buried from the solvent) in the native state would always remain protected. Meanwhile, experimental observations show that amide groups that are relatively buried from the solvent in the native structure are sometimes able to undergo exchange, which means they were able to become exposed to the solvent due to some local unfolding event or conformational change. These small local unfolding events subsequently produce a large number of native conformational states [9, 87, 154].

One major source of experimental data that can provide structural insight on the native state conformational ensemble is available in the protein structure data bank (PDB), where the structure is obtained by Nuclear Magnetic Resonance (NMR) spectroscopy [17, 127, 196]. NMR is one of the two main methods used to determine the protein's structure, which account for about 15 % of all entries in the PDB. On the other hand, majority of other structures, which are solved by X-ray crystallography typically report a single snapshot structure [118].

Unlike X-ray structures, NMR structures are routinely expressed as 'ensembles' of structures, where the entire ensemble represents a possible solution set to the NMR structure determination problem based on experimental data (set of distance and orientation constraints also extracted from an ensemble) [196]. Using the measured NMR constraints, the protein structure can be solved, for example, by an MD-based simulated annealing method such as X-PLOR [17], which generates an ensemble of structures which are consistent with the constraints; typically the best 20 structures (models) are reported in the PDB entry. Presuming that an adequate number of good quality NMR derived constraints are available, the NMR ensemble structure should be well defined and the individual structures (models) will represent some of the plausible populated conformers in the protein's conformational space. The depiction

of the atomic coordinates with multiple sets as represented by NMR structures, rather than a single structure, certainly gives a more realistic image of the proteins structure.

Remark. We should note that very mobile and unstructured regions will often represent more difficulties in NMR structural determination as they may include more errors in the measurements and could have missing constraint data, for example, loop regions are often the least well-defined regions in NMR structures [17]. The extent to which these regions in the ensemble represent the actual flexibility has been debated and it has provided a basis of many research studies [177] and has led to the development of an NMR Validation Task Force [214], a new initiative which should provide additional recommendations in validating and improving the quality of the NMR structures prior to being deposited to the databases. ■

When the protein structure is given as a collection of conformations, particularly the NMR solved structures, the previous FIRST studies would perform the rigidity/flexibility analysis only on a selected single structure (i.e. first structure-model in the NMR ensemble), completely neglecting the information encoded in the other structures of the ensemble. Hence, there would be no distinction in the FIRST algorithmic analysis of X-ray structures with single snapshots and those from NMR solved structures. At present, there has been no better accepted or alternative method suggested in dealing with NMR ensembles.

By selecting only a single NMR structure, not only is the crucial ensemble information omitted, but the development of the constraint network purely from the single structure can make the flexibility analysis more sensitive to the quality of the structure (snapshot) selected (see comments on hydrogen bonds below), and in turn influence the FIRST outcome [109]. It is known that subtle structural variations in the constraint network (i.e. breaking of a few hydrogen bonds and in some extreme cases a single hydrogen bond) can possibly lead to a profound effect on the rigid

cluster decomposition, where a single rigid cluster can break into numerous smaller rigid regions [58, 147, 200].

It is reasonable to hypothesize that analyzing multiple snapshots (when data is available) should alleviate the sensitivities and these deficiencies of the FIRST snapshot analysis, in particular the dependence on the selection of modelled non-covalent interactions in the constraint network, especially hydrogen bonding interactions. We anticipate that work with ensembles should lead to more robust results. Clearly, better ways of predicting rigidity of conformational ensemble than the current FIRST model are desirable.

With this context in mind, one of the main goals of this chapter is to extend the FIRST method and introduce a new FIRST ensemble based method for predicting rigidity/flexibility of protein whose structures is represented as an ensemble. This will be done by averaging the hydrogen bonding interactions over all the individuals structures (NMR models) of the ensemble, and by a further refinement of the mathematical model of hydrogen bonds, to give a single ensemble FIRST prediction; details are provided in the next section. Our objective is to illustrate that the FIRST ensemble adapted predictions can overcome some of the limitations of the traditional single (static) snapshot analysis, and should provides us with a more sensible and improved rigidity/flexibility predictions. In addition to the NMR solved structures, the techniques and methods suggested in this paper can also be applied to an ensemble of snapshots generated by other techniques, for instance conformers generated by MD simulations [67, 122].

To test out our method we will apply it on the native structural ensemble of Acylphosphatase from hyperthermophile *Sulfolobus solfataricus* (Sso AcP) whose ensemble structure (models) has been determined with NMR experiment (see Figure 6.1). Sso AcP is a 101-residue protein, which belongs to the acylophosphatase-like

structural family [12]. In addition to having an NMR solved structure, the hyperthermophile nature of Sso AcP (i.e. which typically do not function at room temperature and normally have enhanced structural rigidity [30, 147]) offers a unique opportunity to apply the rigidity/FIRST analysis to an enzyme that is expected to be non-functional at room temperature due to rigidity. We will compute both the traditional FIRST single snapshot rigidity prediction and our modified FIRST prediction over the entire ensemble. Our FIRST-ensemble predictions will be compared with independently obtained site specific hydrogen/deuterium exchange (HD-exchange) experimental data on Sso AcP [176, 210]. HD-exchange provides very useful data to compare and validated our predictions of rigidity on an ensemble, as the experiment presents measurements from the ensemble.

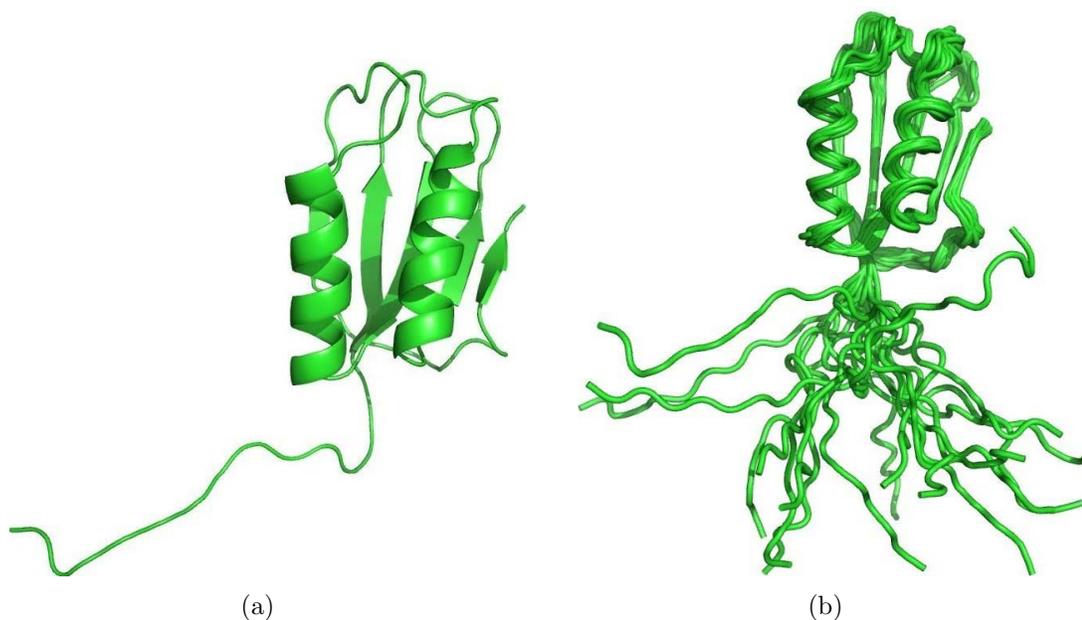


Figure 6.1: 3-dimensional structure of Acylphosphatase from hyperthermophile *Sulfolobus solfataricus* (Sso AcP) (pdb id: 1y9o). This structure is solved with NMR spectroscopy techniques, in (a) the first model (i.e. “most representative structure”) is shown in the cartoon representation, and in (b) the entire structural ensemble (20 models) is shown in ribbon representation (images generated with Pymol [143]).

The second main goal of this chapter is to use our FIRST-ensemble adapted predictions together with other computationally generated data (see below) to give a prediction of Hydrogen-deuterium exchange.

6.2.2 Hydrogen/Deuterium exchange is dependent on Rigidity and Solvent Accessibility

Hydrogen-deuterium exchange (HDX) is a powerful experimental technique as it can provide us with direct information about protein's dynamics and structural stability, and can be used to study protein folding process [9, 15, 42]. It is particularly useful as it is sensitive to the entire structural ensemble, even to the rarest sampled high energy conformers [211].

HDX is a process in which amide protons on the polypeptide 'backbone' and 'exchangeable' protons on some side chains undergo base-catalyzed exchange with solvent [41]. Experimental measurements of HDX are typically confined to the backbone amide protons, which are involved in hydrogen bonding for the maintenance of secondary structure (i.e. α -helices, β -sheets). In short, HDX occurs when the backbone (i.e. main-chain) amide and the carbonyl groups that are hydrogen bonded move apart enough (as a result of backbone flexibility), and in the process break or weaken the hydrogen bond, so that deuterated water can interact and the shared proton is replaced by deuterium (heavy hydrogen). Since the hydrogen bond must first 'break' in order for exchange to occur, HDX can be used to provide a semi-quantitative measure of local thermodynamic stability in secondary structures. HDX can also occur when the amide hydrogens become exposed (accessible) to the solvent [42]. Thus, the observed rate of backbone amide exchange is a function of hydrogen bond strength and solvent accessibility [41, 213] (for more on HDX see [9, 42, 213]).

Using NMR techniques it is possible to monitor backbone amide proton signals as a function of time after exposure to deuterated water (D_2O) (the signal disappears in standard experiments once proton is replaced by deuterium), and assign ensemble exchange rates to individual residues (backbone amide protons) [42]. Residue-specific HDX experimental profile can be obtained, so that residues can be classified into a range of categories depending on how fast they exchange, ranging from very slow exchangers to very fast exchangers [210] (see later section on HDX profile on Sso AcP). Regions that do not exchange or are very slow exchangers are either in very stable and rigid ¹ regions of a protein with strong hydrogen bonding interactions or well buried away from the solvent.

Although experimental methods for measuring hydrogen-deuterium exchange are well established (using NMR and Mass Spectrometry technologies), these techniques can be very costly and time demanding [42]. On the other hand, predicting HDX computationally is not as widely studied, and only a few methods have been suggested [39, 120]. Devising new fast algorithms that can give rapid predictions of the regions in the protein that are protected (or not) from hydrogen-deuterium exchange would be valuable. For instance, predictions of HDX (together with sequence homology) can be used to predict 3-dimensional structures of unknown proteins [120]. By bypassing the bottlenecks associated with the slow experimental techniques, fast computational HDX prediction algorithms can enable high throughput analysis, facilitating the initial analysis of many proteins.

As HDX depends on two parameters, namely structural stability and solvent accessibility [41, 213], we hypothesize that combining rigidity predictions with solvent

¹Note that certain terminology, such as ‘stability’ and ‘rigidity’, can have different meanings in different communities. ‘Stability’ of a protein can, for instance, be interpreted as a region with a well-defined secondary structure, or a region which is unable to undergo HDX. However, this same region could be flexible, since a flexible region may not be able to undergo exchange due to solvent inaccessibility. In this context, ‘stability’ may not imply ‘rigidity’ as is used in the applied rigidity (i.e. FIRST) community, but rigidity should imply stability.

accessibility data should provide us direct insight into regions that are most likely to be protected (or not) from HD exchange. More specifically, if a region of a protein is rigid (particular over the ensemble) it will have a sufficient number of constraints (a rich hydrogen bonding network) to prevent the amide protons from undergoing HDX. That is the backbone amide (NH) hydrogens and its acceptor atoms should remain constrained with sufficiently strong hydrogen bonds, so that its backbone amide protons will not be replaced with deuterium. Similarly, regions whose backbone amides (NH) are buried (inaccessible) from the solvent should not be good exchangers. On the other hand, we anticipate that the backbone amides on the regions that are both flexible and solvent accessible should be good exchangers.

In order to test out this hypothesis, we will introduce a new computational method for predicting HDX, which combines our FIRST-ensemble rigidity predictions with an ensemble solvent accessibility data (obtained from an external program, see details below) of the backbone amide (NH).

This approach will be applied on protein Sso AcP. The predictions we obtain are compared with the experimentally determined HDX measurements of Sso AcP. To our best knowledge, this is the first analysis which uses both rigidity and solvent accessibility for predicting HDX. The algorithmic methods developed in this work are general enough that they could be applied and tested on other proteins.

To summarize, the following new contributions are given in this chapter:

- development of an ensemble-based (i.e. using multiple NMR models) FIRST method for predicting protein flexibility/rigidity
- an introduction of a novel computational method which combines FIRST-ensemble rigidity predictions and ensemble solvent accessibility data as a predictor of hydrogen/deuterium exchange (an ensemble measurement);

Both methods are applied on our case study protein Sso AcP, and the predictions are compared with experimentally determined HDX data.

6.3 Methods and Algorithms

The methodology of FIRST was briefly discussed in the first chapter, and much more detailed information on FIRST can be found in many references [48, 67, 82, 83, 84, 109, 122, 200]. In this section we give a detailed description of our approach to using the set of NMR conformations (which is equivalently applicable to say conformers generated by MD simulations [122]) to give a single FIRST ensemble rigidity/flexibility prediction. The emphasis of the extensions of FIRST to ensembles is on the careful modelling and revision of the current FIRST rigidity model of the hydrogen bonds. We also briefly discuss the calculations of the solvent accessibility data, and how we combine it with ensemble rigidity predictions, in order to obtain a fast computational prediction of HDX. All the calculations and predictions are carried out on our case study protein Sso AcP.

6.3.1 FIRST-ensemble rigidity/flexibility predictions

Comments on Hydrogen Bonds and their impact on FIRST predictions

The current rigidity/flexibility predictions using the program FIRST depend solely upon the set of constraints that are included in the multigraph. Once the set of constraints (covalent bonds, hydrogen bonds, hydrophobic interactions) are determined, the set of constraints is modelled as being always present, and all other potential constraints are always excluded. Given this all or nothing constraint representation

of a protein, the well established success of FIRST approach may be initially unexpected. Detailed discussion on how various constraints are modelled can be found in the FIRST literature [109].

The main control on the constraints is specified by a hydrogen bond energy cutoff (preselected by the user), which determines the set of hydrogen bonds to be included as constraints in the graph [109]. The strength of a hydrogen bond is calculated based on its local donor atom–hydrogen–acceptor atom (angular and distance) geometry (stronger interactions are attained when these atoms are nearly collinear, see [82, 109, 200] for details). As the output of FIRST is almost entirely dependant on which set of hydrogen bonds are included (i.e. energy cutoff) in the molecular constraint graph [200], in order to get as accurate FIRST results as possible, it becomes essential that the set of hydrogen bonds that are included in the analysis are carefully modelled in the graph. Revisiting and understanding the current hydrogen bond mathematical rigidity model, and clarifying its strength and limitation will help us consider refinements and will facilitate a meaningful extension of FIRST predictions to structural ensembles.

Although individually weaker than covalent bonds (which are always present in FIRST), the combined effects of hydrogen bonding is crucial in maintaining the stability of proteins. They are particularly important as they stabilize key secondary structural elements, the α -helices and β -sheets [25]. In Figure 6.2, backbone hydrogen bonds between a NH donor and a CO group are shown in a typical α -helix and between two strands of a β -sheet. Hydrogen bonds between side chain atoms can also act as important constraints, holding distinct regions (distant in the sequence) in the protein together, and these play important roles in establishment of tertiary and quaternary structures that are found in proteins. Hydrogen bonds have also been observed in

the active sites of many proteins [148, 215], whose breaking or forming can help turn a protein from active to inactive state.

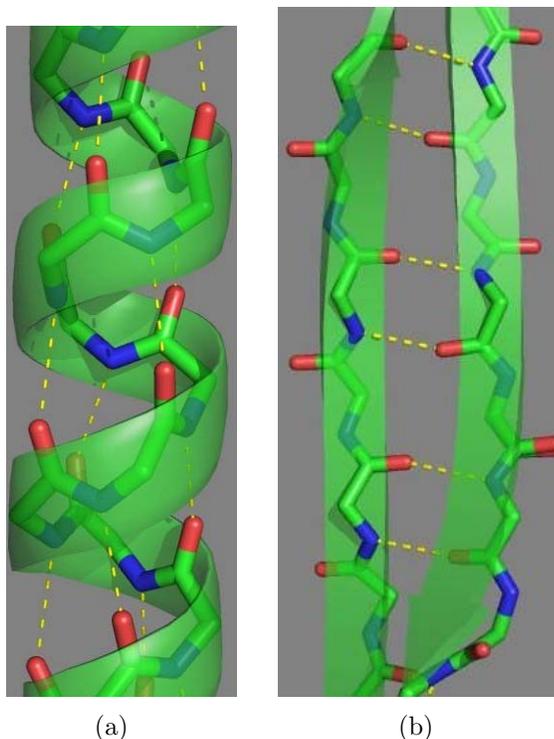


Figure 6.2: Two main secondary structure elements, α -helix (a) and β -sheets (b). Hydrogen bonds between backbone NH and CO groups, which stabilizes these structures are indicated as dashed lines. Unlike α -helices, β -sheets are constructed with multiple strands of polypeptide chain, where hydrogen bonding between the NH and CO groups occurs between two different strands. Generated with Pymol.

Using a hydrogen bond energy cutoff, FIRST will distinguish between weak and strong hydrogen bonds. However, no further quantitative distinction is made. In other words, any two hydrogen bonds that meet this threshold are included as constraints in the graph, and will remove the same degrees of freedom [109, 200]. In fact, in terms of the mathematical model of rigidity, those hydrogen bonds that are included as part of constraints are modelled equivalently to the covalent bonds (i.e. using five bars (edges) between the hydrogen and the acceptor atom). To be able to distinguish weak hydrogen bonds (poor geometry) from the strong hydrogen bonds

(good geometry) is very useful and is an important modelling tool that has made FIRST successful on the analysis of proteins using a single snapshot (conformation). However, there are further uncertainties which are unique to hydrogen bonds that need to be considered.

Unlike covalent bonds, it is common for hydrogen bonds to spontaneously break and reform (flicker) to adapt to the changing conformation of atoms induced by the internal mobility and flexibility. Dynamics simulations have confirmed that some hydrogen bonds have very short life times and undergo ‘flickering’ on the order of tens of picoseconds to a few nanoseconds [122, 170]. It is also possible for strong hydrogen bonds to become unstable (i.e. at some snapshot it can break) over time [122]. In [200], the authors have indicated that small variations in the donor-acceptor distance can lead to significant changes in the hydrogen bond energy strengths. Each conformation that is sampled under native conditions has a specific set of hydrogen bonds, which could slightly change for a different conformation due to the local changes in geometry. The changes would be particularly evident in the most unstable and flexible regions of the protein.

In [122] it was reported that FIRST runs on different snapshots of proteins generated with molecular dynamics simulation display differences in the rigid cluster decompositions. The changes in the FIRST predictions from snapshot to snapshot are due to local changes in geometry which can cause FIRST to identify different sets of hydrogen bonds. This could cause enough changes in the constructed constraint multi-graph that potentially alter the rigid clusters. Because of the ‘flickering’ (time-dependant) nature of hydrogen bonds, in addition to the strengths of hydrogen bonds, the authors of this study incorporated the lifetimes of hydrogen bonds. More specifically, they include the hydrogen bonds as constraints in FIRST, if they were

present for a high fraction of time over the molecular dynamics simulation. This approach gave an improved FIRST prediction which better matched the experimental evidence and molecular dynamics simulations. We anticipated that FIRST predictions on snapshots from NMR ensembles may also show enough variations, which will be investigated shortly. Hence, further refinement of the hydrogen bond models, whenever a collection of snapshots are analyzed is crucial and needs to be further investigated.

As outlined in the next subsection, and tested on our case study of NMR structure of Sso AcP, in order to give a single FIRST ensemble prediction, a natural first step is to obtain an average hydrogen bond strength for each hydrogen bond over all NMR conformers. Only hydrogen bonds that have strong average strength (energies) over the entire ensemble will be included as constraints in the ensemble graph (see details below and in Algorithm 6.3.1).

In addition to determining the average strength of every individual hydrogen bond, unlike any previous FIRST studies, we will also apply modifications to the rigidity model of hydrogen bonds. One such particular refinement that we will investigate is adjusting the number of bars used in hydrogen bonds. More specifically, we will allow the number of bars (edges) to vary between hydrogen and acceptor atom between 1 and 5 (see Figures 6.3 and 6.4). With this additional step we will attain a more accurate representation of hydrogen bonds over the entire ensemble.

This refinement and fine tuning of the mathematical model of hydrogen bonds will mainly be used as a penalty for any hydrogen bond that is determined by FIRST to be very weak (i.e. lacks good geometry, so it is not modelled as a constraint) in some of the NMR models. In other words, if the hydrogen bond does not persist over all snapshots, it will be subject to a penalty. Hydrogen bonds that are relatively

strong in most (see details in next section) NMR models will not be subjected to a decrease (i.e. penalty) in the number of bars.

The fine tuning and adjustments of hydrogen bonds via a different number of bars allows us to gradually weaken those bonds that may not persist in all the snapshots (models). If the average energy strength of a particular hydrogen bond is sufficiently high over the entire ensemble, yet it is not present (strong) in majority of snapshots, this bond will still be included as a constraint. However, given that there will be a bar(s) penalty for this kind of hydrogen bond (which reflects the lack of persistence over the ensemble), a constraint with say 2 or 3 bars will remove less DOF from the overall system than the traditional 5 bar constraint.

This type of revised modelling of hydrogen bonds allows us to incorporate the information from the entire ensemble, and it facilitates our move away from the simple ‘on/off’ type of modelling of hydrogen bonding constraints that has been previously used in all FIRST studies. We hypothesize that this should generally prove to be very beneficial in obtaining FIRST ensemble predictions.

As this new rigidity modelling can provide much more fine-tuned information about hydrogen bonds, it may also help identify with higher certainty than a single snapshot analysis, when hydrogen bonds are essential to the maintenance of the rigidity in protein ensembles. Due to the combinatorial nature of the underlying rigidity theory behind FIRST and the greedy nature of the pebble game algorithm [207], varying the number of bars will not alter the computational speed and efficiency of the FIRST predictions. In the next subsection we offer detailed algorithmic descriptions of these extensions.

Remark. We have mainly focused our discussion on the hydrogen bonds, and their impact on the FIRST results. In addition, hydrophobic contacts or tethers, which are modelled as constraints (i.e. 2 edges) between carbon and/or sulphur atoms in

close proximity [67, 82], are also critical to protein flexibility and FIRST predictions [67]. However, unlike hydrogen bonds, hydrophobic tethers are modelled as fixed constraints. It is reported that this is because the strength of hydrophobic interactions in the modelling remains constant and can even become stronger as the temperature increases (see discussion in [109] for details). Given the way hydrophobic tethers are modelled in FIRST (i.e. using 2 bars/edges which removes maximum 2 DOF), and because of the non-specific nature of hydrophobic interactions, their influence on FIRST results will not be as strong as that of hydrogen bonds [122]. Studies suggest that only preservation of the total number of identified contacts (i.e. constraints) is important in the FIRST analysis [122], and the fact that they could very often be switching partners (as is the case in very tightly packed groups) should not be crucial in FIRST analysis.

We postulate that the hydrophobic interactions should not exhibit the same type of sensitivities as hydrogen bonds to the conformational variation that can occur among the NMR conformers, and hence they should not significantly alter the FIRST ensemble predictions. This will be further probed in the results and discussion section.



6.3.2 FIRST-ensemble algorithm

We now describe the procedure that gives a single FIRST prediction (rigid cluster decomposition) on the entire ensemble from an NMR protein structure. We call such predictions *FIRST-ensemble*. Typically a NMR file will contain 20 models that best fit the NMR data. So we assume that $m = 20$ in this description. The procedure is easily adapted to a file that contains a different number of structures.

Algorithm 6.3.1 – *FIRST-ensemble procedure:*

Input: NMR PDB file.

Output: single FIRST rigidity/flexibility ensemble prediction (i.e. rigid cluster decomposition).

(1.) Run FIRST on every NMR model m ($m = 1..20$) using a hydrogen bond energy cutoff of 0 kcal/mol and obtain the energy strength E_i^m for every hydrogen bond i . (Typically this can be retrieved from `hbounds.out` file, see FIRST manual).²

(2.) Calculate the average energy strength E_{AVGi} for each hydrogen bond i over all 20 models:

$$E_{AVGi} = \frac{\sum_{m=0}^{20} E_i^m}{20}$$

(3.) Use the following criteria to assign the bar (edge) penalty between hydrogen and acceptor atoms for each hydrogen bond i :

(a) Let hydrogen bond i be considered present in a specific model (snapshot) if its energy strength is less than (more favourable than) -0.5 kcal/mol. Denote the total number of times hydrogen bond i is present out of 20 models as N_i .

(b) Assign the appropriate number of bars (edges) B_i (correction factor) for every hydrogen bond i , using the following rule:

$$B_i = \lceil \frac{N_i}{20} \times 5 \rceil \text{ (where } \lceil x \rceil \text{ is a ceiling function, see Figure 6.4)}$$

²For efficiency, note that actual rigidity (pebble game) analysis in this step is not needed to be performed, as we are only extracting the energy strengths of hydrogen bonds.

(4.) Select the first NMR model (any model could be selected - see discussion and results) and run FIRST³ on this structure using hydrogen bonds from 3. (a) with input E_{AVG_i} and B_i .⁴

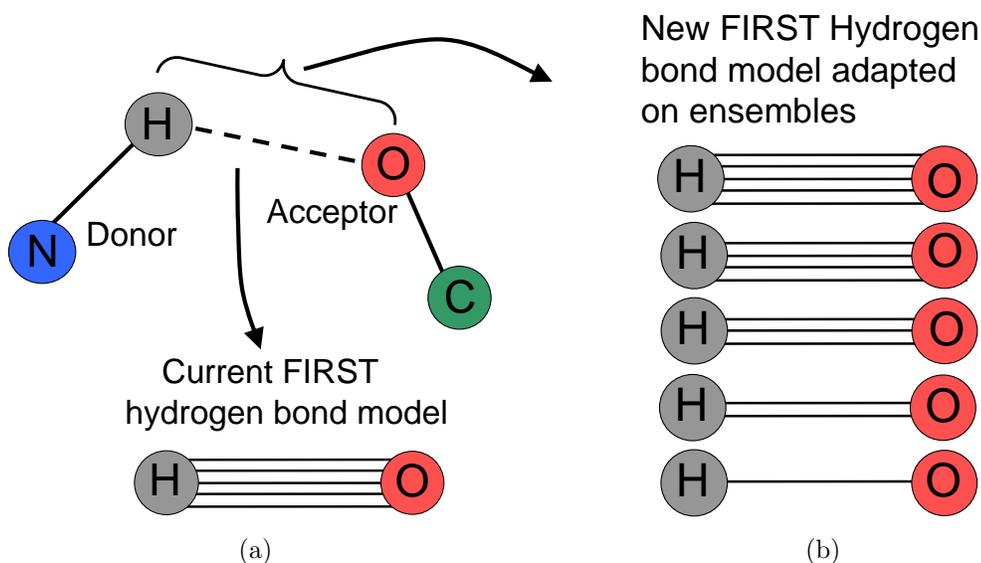


Figure 6.3: In previous studies using FIRST, hydrogen bonds would be modelled with 5 bars (edges, constraints) between hydrogen and acceptor atoms (a). In the FIRST-ensemble predictions we allow the number of bars to vary between 1 and 5, depending on the persistence of a hydrogen bond over all structures in the ensemble (b).

In step 1. we chose to run FIRST with energy cutoff of 0 kcal/mol as this cutoff would consider all potential hydrogen bonds (i.e. weak and strong). Note that hydrogen bond i gets 5 bars (correction factor $B_i = 5$, i.e. no penalty) if it is present in 17 or more models, four bars ($B_i = 4$, i.e. one bar penalty) if present in 13 to 16

³Recall that prior to running FIRST, a hydrogen bond energy cutoff has to be selected. In all the FIRST predictions which are performed on both the individual NMR snapshots of Sso AcP and on the single ensemble prediction (as outlined in this algorithm), we will use the -1.0 kcal/mol hydrogen bond energy cutoff (this is a typical default recommended cutoff, see FIRST user manual [48]) (i.e. it includes only those hydrogen bonds whose energy is less than (i.e. more favourable than) -1.0 kcal/mol (i.e. strong hydrogen bonds), and avoids all weaker hydrogen bonds (i.e. > -1 kcal/mol).

⁴As FIRST will be looking for a user defined hydrogen bond energy file, the E_{AVG_i} with bar penalties can be placed in the hbonds.in file, see FIRST manual for details.

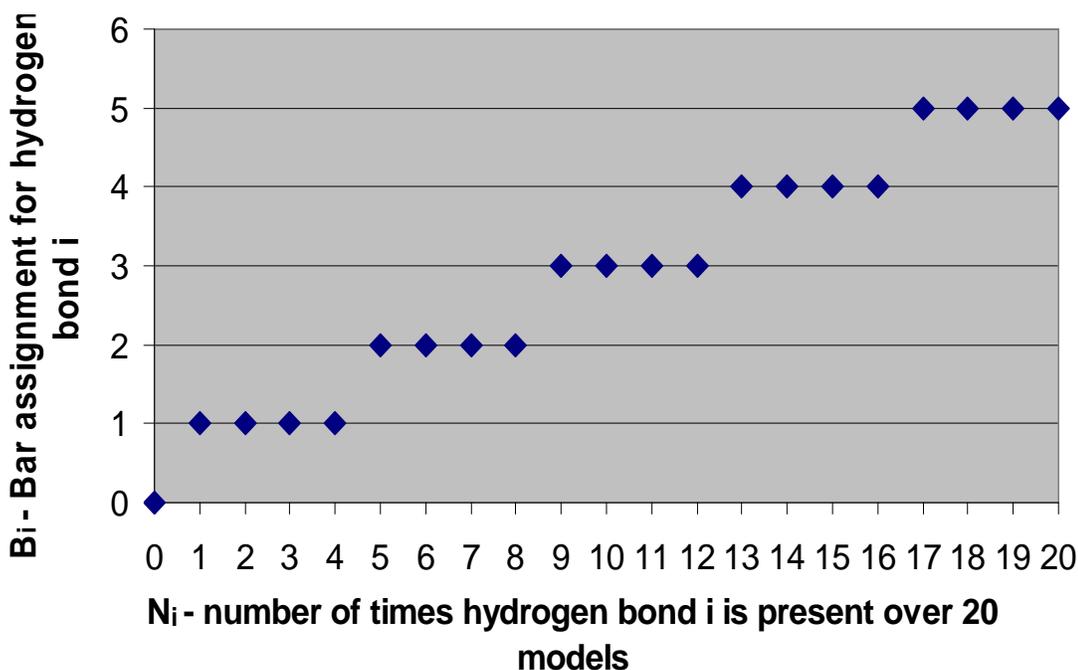


Figure 6.4: Assigning number of bars (edges) B_i for hydrogen bond i , which varies from 1 to 5, depends on how often the bond is present in the models of the ensemble.

models, three bars if present between 9 and 12 models, two bars for 5 to 8 models and only one bar (see remark below about assigning one bar) if it is present in 1 to 4 models (see Figures 6.3 and 6.4).

As an example, consider a hydrogen bond i that is present, for instance in 10 out of 20 models, with relatively strong individual energies E_i^m . The average E_{AVGi} strength may still be relatively (negatively) high, such that it passes the energy cutoff of -1.0 kcal/mol⁵, so this bond would be included as a constraint. The usefulness of correction factor B_i can be seen as it takes into account the absence (weak energy, bad geometry) of this hydrogen bond in the remaining 10 models, giving it a 2 bar penalty, and an assignment of 3 bars. Hence, the correction factor makes this hydrogen bond less constraining (removes less DOF) on average than the traditional 5 bar constraint.

⁵This is a default user cutoff in FIRST, see [48].

In this approach, we still continue to use an energy cutoff to distinguish weak from strong hydrogen bonds (averaged over the ensemble), with the added feature that the correction factor B_i allows us to incorporate the measure of the persistence of a hydrogen bond over the ensemble into FIRST analysis. Of course, if a hydrogen bond is persists in all or most (17 or more) models, it will retain its 5 bars.

Implementation related remark. For completeness, we want to make a small technical remark. In certain special cases, when assigning one bar to the hydrogen bond, a small adjustment might be needed. This occurs when both hydrogen atom and acceptor atom (which is not modelled as connected to another atom through a double bond (or non-rotatable) in FIRST) were one-valent prior to the insertion of the hydrogen bond.

In FIRST one-valent atoms are modelled with 5 DOF (5 pebbles), as rotation around its bond is non-visible in biochemistry (i.e. it just spins around its axis). So, when we assign one bar between the two one-valent atoms, each atoms now becomes a full rigid body with 6 DOF (6 pebbles), and placing one bar (edge) between them removes one of the additional DOF, but leaves an extra DOF (pebble) (i.e. it adds extra flexibility). To visualize this, refer to Figure 6.5. When one bar is inserted (Figure 6.5 (b)), an extra pebble appears. Moreover, note that before one bar was inserted, the hydrogen atom was in the same rigid cluster as the nitrogen atoms, and the oxygen atom is in the same rigid cluster as carbon atom. It is also possible that hydrogen and acceptor (say oxygen) were also part of the same rigid cluster prior to inserting one bar. After one bar is inserted, the hydrogen and the acceptor atom become their own rigid clusters. If desirable, one way to correct this would be to assign an additional bar (edge) for this bond, so it gets two bars.

However, we feel that this is not that essential as all the other rigid clusters remain unchanged (for instance nitrogen and carbon in Figure 6.5 are part of the

same rigid cluster before and after one bar is inserted). The reasoning is basically the same as why having an extra free pebble on a one-valent vertex will not have an effect on the rigid clusters. This can be easily mathematically verified in terms of the pebble game properties, but is not important here. It would also be easier in implementation sense to not start adding an extra bar.

We are mainly addressing this minor technical adjustment in order to keep consistent with how one valent atoms are modelled in FIRST, and for any future work considerations of implementation of the FIRST-ensemble algorithm. We suspect that hydrogen bonds that get one bar should also be very rare, as they still have to pass the average energy cutoff threshold (for instance, in our case study protein below, none were found). In other words, it is unlikely that in four or less structures there are very strong hydrogen bonds, given that in remaining 16 or more models these bonds are very weak. ■

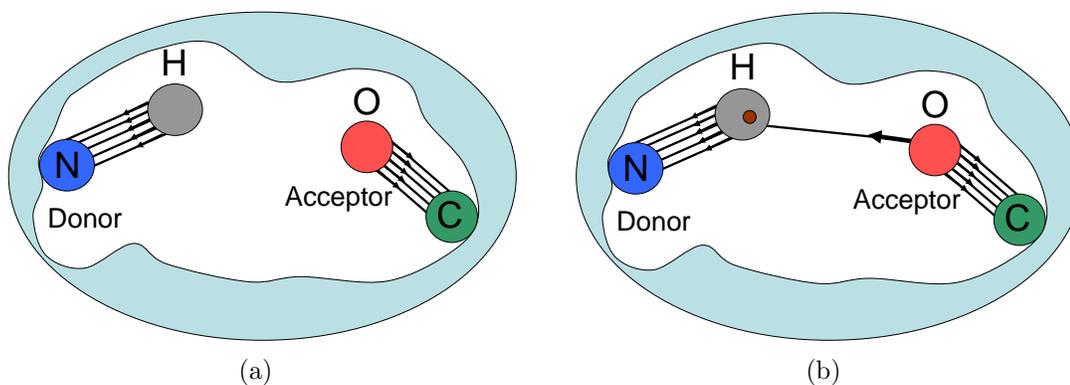


Figure 6.5: In FIRST one-valent atoms are modelled with 5 DOF (5 pebbles), as the rotation around its bond is not visible. Note that both hydrogen and acceptor have 5 outgoing edges and no free pebble. If both hydrogen and acceptor atoms are one-valent prior to insertion of one bar (a), then adding one bar between hydrogen and acceptor increases flexibility (an extra free pebble appears) (b), as both hydrogen and acceptor are now treated as having 6 DOF (6 pebbles). In this case an extra bar can be added between hydrogen and acceptor (i.e. so there is 2 total bars). Alternatively, one bar can remain as rigid clusters are essentially not altered.

6.3.3 Solvent Accessibility Calculations

As we will be combining rigidity with solvent accessibility to probe hydrogen-deuterium exchange (HDX), we now discuss how we obtain the ensemble solvent accessibility data. Given that the experimental HDX on Sso AcP measures exclusively the HDX rates of backbone amide protons, we are only interested in obtaining solvent accessibility of the backbone amides (NH).

The solvent ‘accessible surface area’ (ASA), was initially introduced by Lee and Richards [116] to quantitatively represent the extent to which atoms on the protein surface are able to make contact with water. ASA is usually defined as the surface (measured in square angstroms) traced out by the center of a water sphere (having a radius of about 1.4 angstroms) as it is rolled over the surface of the protein (van der Waals surface) [24, 195]. It is a standard practice to convert the ASA to a normalized form, ‘relative solvent accessibility’ (RSA) [153, 195]. RSA is a very useful measure of solvent accessibility and can be quantified as a percentage of accessibility (from 0% which is completely buried (inaccessible) to 100% accessible). RSA represents the ratio of ASA in the current folded conformation (pdb snapshot) to the ASA in the extended-unfolded conformation. The extended state of residue X is usually taken in the Gly-X-Gly tripeptide state (due to Glycine’s small side chain group containing only a single hydrogen), and is used to quantify the maximum ASA in the extended conformation.⁶ For terminal residues X, the extended state is usually taken in X-Gly dipeptide state. Several methods have been developed for the prediction of RSA [1, 86, 171, 195].

In this work, we use the WHATIF [195] server to determine the accessibility (RSA) of backbone nitrogens (hydrogens are usually excluded and only heavy atoms are used in the computational methods that predict solvent accessibility). This will

⁶We can also think of RSA as the percentage of the accessibility in the unfolded state, still available in the folded protein [195].

be performed on our case study protein Sso AcP. We use the WHATIF accessibility (RSA) calculation data from every structure of an ensemble, and then calculate the average (mean) RSA of every backbone nitrogen (averaged over over all 20 models of an ensemble). Accessibility (RSA) of backbone amides (NH) have been used previously as a means to explore HDX rates [169].

We are primarily interested in identifying the set of residues whose (ensemble averaged) backbone amides (NH) are completely or almost completely buried from the solvent, as these residues should not be undergoing exchange. It is a common practice to project the accessibility into two states (buried/exposed) or three or more states [153]. There is no clear or consistently used threshold value to distinguish buried (inaccessible) from solvent-accessible residues (or atoms) [2, 153]. The reported RSA cutoffs to distinguish buried states in the literature have varied greatly, with values such as 0%, 4%, 5%, 6%, 7%, 9% and 10%, etc. [25, 86, 119, 129, 153].

In our work, we define any residue as *almost completely buried* (or *solvent inaccessible*) if its backbone nitrogen RSA value, averaged over an entire ensemble, is less than 4%, otherwise they are defined as *exposed* (or *solvent accessible*). This strong cutoff allows us to select only those residues whose backbone amide (NH) are closest to being completely buried. A smaller accessibility cutoff was not chosen, for instance average 0% RSA, due to potential numerical rounding approximations, and to eliminate the situations where there is slight increase in average accessibility as a result of potentially higher accessibility in one or two outliers (models).

When visualizing accessibility on a protein's 3D structure, almost completely buried residues will be coloured blue. For completeness and richer visual display, we further classify the exposed residues in the following categories (whose thresholds are somewhat arbitrarily selected but close to [153] for example): if the backbone nitrogen has accessibility which is between 4 % and 10 % we say that the corresponding

residue is *somewhat exposed* (yellow), between 11 and 30 % *mostly exposed* (orange), and greater than 30 % *almost completely exposed* (red) (see Figure 6.13 in the next section). The colour coding for these remaining categories are assigned for easier visualization, which should aid in discussion and comparison with the experimentally determined HDX. The results that are reported (see next section) on Sso AcP are not sensitive to the threshold cutoffs.

6.3.4 Combining FIRST-ensemble and Solvent Accessibility on ensemble as a Predictor of HD-exchange

As was described in the introduction, we have hypothesized that combining rigidity/flexibility (obtained by FIRST-ensemble Algorithm 6.3.1) and solvent accessibility predictions (also from an ensemble) should give us a reasonable approximation for differentiating the regions in the protein (backbone NH) that are protected from exchange from those regions that will likely undergo exchange.

Both rigidity/flexibility and solvent accessibility hold valuable information that can assist in predicting HDX. However, on their own they would not be good predictors. Consider a scenario where a given region of interest in the protein is flexible but its amide protons are buried away from the solvent, which would make this region a slow HD exchanger. Thus, if we did not have the solvent accessibility information, sole rigidity/flexibility predictions would not tell us about the lack of (i.e. slow) HDX in this region. To keep the backbone amide protons buried in a flexible regions, it would be reasonable to assume that any potential motions that protein undergoes in that region would be composed of small local motions, as large scale non-trivial motions could potentially expose the backbone amide protons to the solvent. In contrast, consider a rigid region which is accessible to the solvent. Using only solvent accessibility in this case is insufficient to probe HDX, since in a typical rigid region of

a protein there will be significant number of hydrogen bonding interactions that will restrict the motion of the backbone amide protons and acceptors. Another difficulty with using accessibility as a stand-alone predictor would occur with α -helices, as many α -helices of protein 3D structures are amphipathic [15, 119], meaning that one side of the helix will mostly be composed of hydrophobic residues which are orientated toward the protein core, and the other side is hydrophilic and more exposed to the solvent (of course there are exceptions [15]). Even if a helix was mostly exposed to the solvent, and yet it is well structured and retains many of its hydrogen bonds, which would come up as rigid in FIRST-ensemble, it would still be a slow exchanger (i.e. well protected).

By incorporating both rigidity and accessibility, we are able to propose a general method for predicting HDX, which should eliminate many difficulties of using solely rigidity or accessibility.

We reformulate our initial hypothesis as follows: if a region is rigid and/or solvent inaccessible (buried), its backbone amide protons should be protected from HDX. In other words, exchange should normally occur, when the protein is both flexible and its backbone amide protons are solvent exposed. Following this hypothesis, we are now ready to outline a procedure that combines the FIRST-ensemble rigidity predictions and solvent accessibility ensemble data, as a measure of HDX (which is an ensemble based experiment).

Algorithm 6.3.2 – *Rigidity and Accessibility as a prediction of HDX*

Input: NMR PDB file.

Output: Prediction of regions that will most likely be protected from HDX (i.e. no exchange or slow exchangers), and regions which are likely to undergo exchange.

- (1.) *Run the FIRST-ensemble procedure (Algorithm 6.3.1) to obtain the rigid cluster decomposition for the ensemble.⁷*
- (2.) *Consider rigid clusters from 1. and combine into a single coloured rigidity-region. Display this combined rigidity-region with a unique colour on the protein's 3-dimensional structure, and choose a different colour for the remaining flexible regions. (We will colour the combined rigidity-region with the same colour as the largest rigid cluster, which is blue by default, and intervening flexible regions with gray.)*
- (3.) *Superimpose the average solvent accessibility regions (preferably using the colouring convention given in Section 6.4.4) on the remaining flexible regions (excluding α -helices and prolines) of the protein from step 2. 'Almost-completely buried' (inaccessible) regions should be displayed with the same colour as the combined-rigidity region (blue) in step 2.*

The combined rigidity-region and almost-completely buried (inaccessible) region (both coloured with blue) correspond to the regions that we predict will most likely be protected from HDX (i.e. slowest exchangers). The remaining regions: flexible (gray), somewhat exposed (yellow), mostly exposed (orange), and almost completely exposed (red) are the regions most likely not protected from undergoing HDX (i.e. fastest exchangers).

Note that an equivalent combined colouring in step 3. can be obtained if we superimpose the solvent accessibility on the whole protein including rigid and flexible

⁷Note that each atom will belong to a unique rigid cluster. By default rigid clusters in FIRST are typically displayed if they contain at least 20 atoms. We will also only consider these rigid clusters. So flexible regions of the protein will usually be composed of many small rigid clusters (in most cases as small as single atoms).

regions (excluding α -helices), however, superimposing the accessibility only on the flexible regions can technically reduce the amounts of solvent accessibility calculations. Due to the amphipathic character of α -helices (see Section 6.4.4), solvent accessibility is excluded for α -helices. Solvent accessibility is also not used for prolines, as HDX is not assigned to prolines, since amino acid proline has no backbone amide hydrogen.

6.4 Results and Discussions

In this section we apply the methods and algorithms from the previous section on the Sso AcP protein. As we will compare the predictions with experimentally determined HDX profile of Sso AcP, we begin this section with the discussion of the experimentally obtained results.

6.4.1 Experimental HDX profile of Sso AcP

We will not get into any detailed discussion of the experimental HDX results, or how the experiment was performed, as that is beyond the scope of this thesis and is not needed for our purpose (see [176, 210] for details). We will only offer the bare minimum we need so we can compare our computational predictions. As we mentioned earlier this data was originally collected in [210], and also appears in [176].

The cartoon representation of the Sso AcP protein is coloured and labelled according to its secondary structure (as provided by authors of the pdb file) in Figure 6.6. We will use the corresponding numbering of secondary structures that is given in this figure throughout our discussion.

In Figure 6.7 the experimental HDX profile of Sso AcP is overlaid on the 3-dimensional structure. Red and orange residues represent the very fast and fast exchangers, respectively. These faster exchangers appear exclusively in the loops and

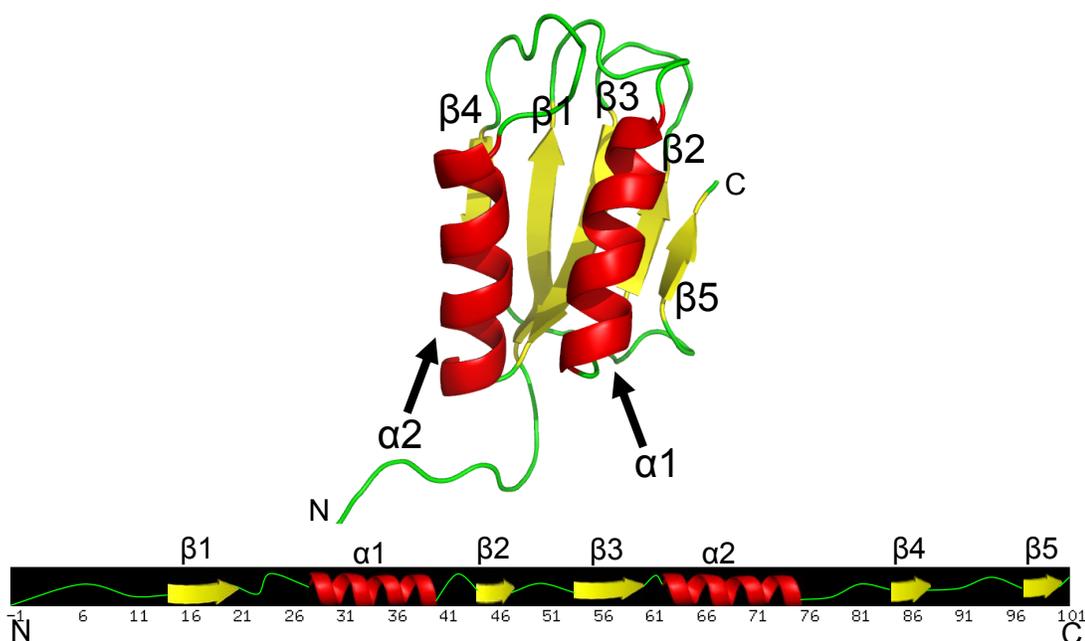


Figure 6.6: Secondary structure representation of Sso AcP (pdb id: 1y9o), and the main-chain representation of the secondary structure. Secondary structures are indicated as defined by the authors in the pdb file.

in the unstructured N-terminus tail region (i.e. first 12 residues) of the protein, with the exception of two residues, one in the N-terminal end of α -helix 2 and in the middle of α -helix 1. Yellow residues are the medium exchangers and blue are the slowest exchangers (i.e. exchange not observed over three weeks after exposure to D₂O [210]). In the gray regions no HDX measurements could be made as NMR signal is not seen (or it is a proline), and most of them are found in unstructured parts of the protein [210]. The notable exception is the cluster of (gray) residues bounded by Gln25 and Lys31, of which good number correspond the N-terminal end of α -helix 1. It is suspected that the N-terminal half of helix 1 is an unstructured and flexible region in the protein, and likely undergoes larger-amplitude ‘molten globule-like’ conformational dynamics [210]. Residues undergoing molten-globule-like conformational dynamics are frequently not detectable by NMR due to extreme broadening of the peaks [211].

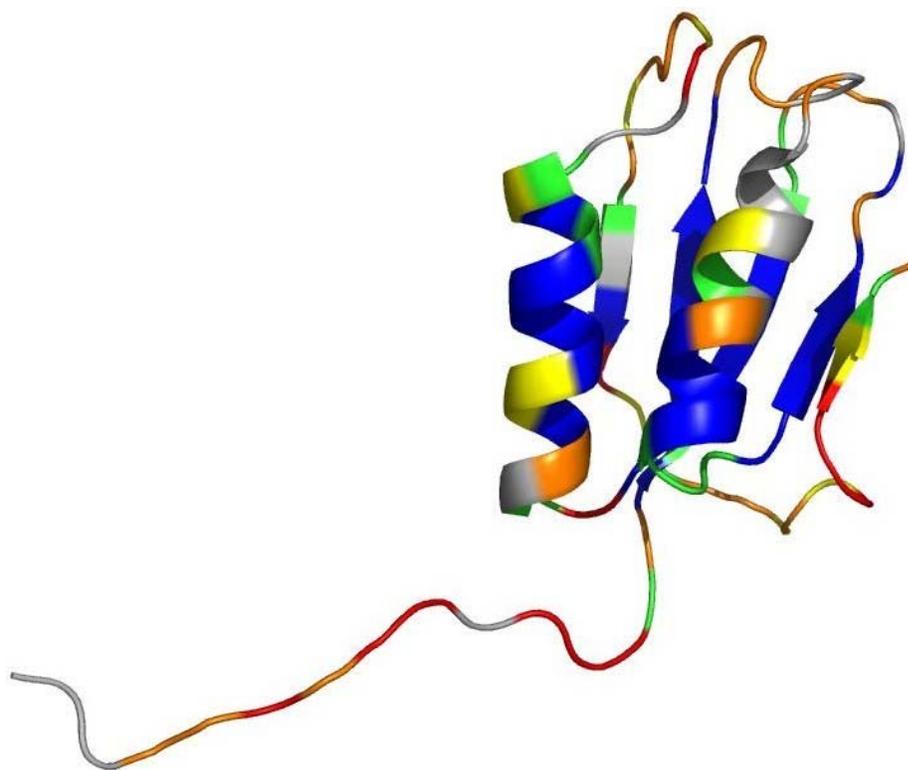


Figure 6.7: Hydrogen-Deuterium Exchange (HDX) profile of Acylphosphatase from *Sulfolobus solfataricus* (Sso AcP) mapped onto its 3-dimensional structure [176, 210]. Blue regions represent the very slow (i.e. most protected) backbone amide exchangers, and are mostly found in the C-terminal half of helix 1, most of helix 2, and the three central strands of the β -sheet (see Figure 6.6). Yellow are the medium exchangers. Green could not be assigned reliable HDX data as NMR signal is too weak [210, 211]. The faster exchangers (orange and red) are mostly found in the unstructured loops and in the N-terminus tail region. Gray residues could not be assigned HDX rates as NMR signal is not seen, and most likely represent unstructured and flexible regions and also prolines [210].

Prolines are also coloured gray as they do not undergo exchange (prolines do not have a backbone amide proton).

Regions of the protein with high exchange rates (faster exchangers - red, orange) correspond to the least protected regions from undergoing exchange, and regions with slowest exchange rates (blue) are the most protected regions in the protein from undergoing exchange. The slowest exchangers are mostly found in the C-terminal half

of helix 1, most of helix 2, and three central antiparallel β -strands (strands 1, 2 and 3) of the β -sheet. α -helix 2 has significantly more slow exchangers than α -helix 1. The two halves of α -helix 1 seems to be significantly different. Unlike the N-terminal half which seems very unstructured, the C-terminal half has only slow exchangers and is very well protected from undergoing exchange. For more details see [176, 210].

We should mention that this experimentally obtained data is rare (i.e. similar data is difficult to obtain in the literature), as it reports a high coverage native (ensemble) state HDX profile (i.e. most residues are labelled), and on top of that the protein has a solved NMR structure [210, 211]. Moreover, this data is particularly interesting given the hyperthermophile ⁸ nature of the protein.

6.4.2 FIRST rigidity/flexibility predictions using individual NMR models

As we mentioned earlier, the standard practice in analyzing the rigidity/flexibility of NMR structures using FIRST is to only pick a single (usually first model) structure. Before we present our FIRST-ensemble predictions on Sso AcP, we will actually perform the regular ‘single snapshot’ FIRST analysis on all 20 snapshots (NMR models) of Sso AcP (pdb: 1y9o). The rigid cluster decomposition of the first ten models is displayed in Figure 6.8. The flexible and rigid regions were obtained in about a second of computational time per structure.

There are just two significantly large rigid clusters (recall that only rigid clusters containing at least 20 atoms are displayed), corresponding to the regions (blue and green) of the two α -helices; gray regions are flexible regions. In the first model (so called most representative structure), which is typically the selected conformer in the previous FIRST studies on NMR structures, both helices are almost completely

⁸Hyperthermophile organisms live in extremely hot environments.

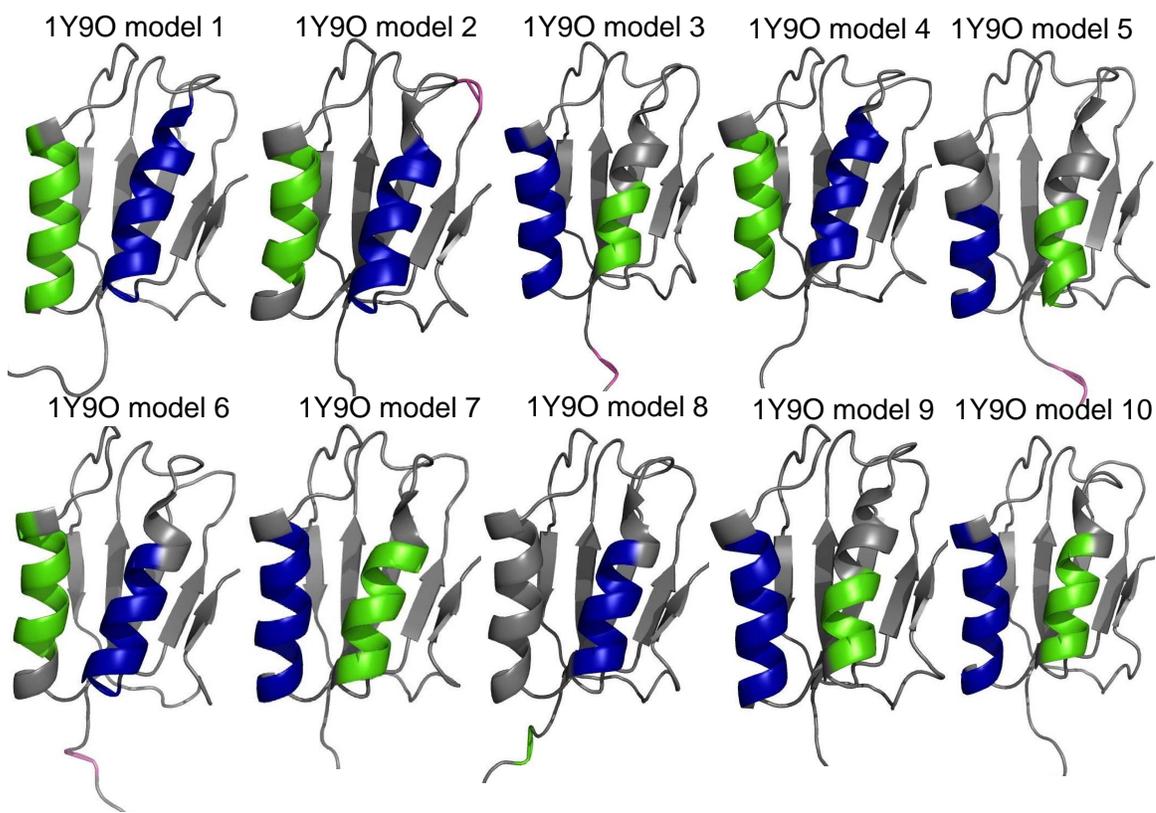


Figure 6.8: Rigid cluster decomposition using FIRST on NMR models (structures) of Sso AcP (same energy cutoff of -1.0 kcal/mol is used in all structures). The output of only first 10 models from the ensemble is displayed. The blue rigid cluster is the largest rigid cluster, followed by the green rigid cluster. Note the variation in the size of the rigid clusters among the NMR models.

rigid. The important observation is that in many other models there is a substantial difference in the size of these two rigid clusters. The blue rigid cluster is always the largest rigid cluster, and we can observe that there is a switch in which helix gets declared as blue or green. In some models, parts of, or even an entire helix (model 8) are flexible. This shift is particularly evident in the ends of helices, notably in the N-terminal end of α -helix 1, which has significantly weaker hydrogen bonding interactions than its C-terminal end. In fact, in many models (for instance 2, 3, 5, 6, 7, ... etc.), it becomes obvious by visual inspection that the N-terminal end of helix

1 is unravelling and is very unstructured, and it more closely resembles a random coil/loop than a helical structure.

It is known that small changes in conformation of atoms can lead to altered hydrogen bonding geometry (hydrogen bond energy values) (see [200]) and a consequent change in the total number of included hydrogen bonds in FIRST. Hence, the variation in the size of the rigid clusters across the different NMR models that we have encountered is not surprising. The number of strong hydrogen bonds (i.e. with energies less than -1 kcal/mol) among the 20 models of Sso AcP ranges from as few as 43 hydrogen bonds to as many as 54 hydrogen bonds. These differences will clearly have an effect on the total number of remaining DOF from model to model, and a potential effect on the size of rigid clusters, particularly when the change in the number of hydrogen bonds occurs within a rigid cluster. These observations are similar in flavour to the study by Wells et al. [200], where the authors found that FIRST analysis on structurally similar proteins from different crystal structures (crystallized under different conditions) with the same energy cutoff can produce different rigid cluster decompositions. The slight structural heterogeneity among the models and the subsequent conformational variations of atoms across the ensemble can cause a slight change in the geometry of hydrogen atom - donor - acceptor atom, illustrating the sensitivities and dependence of the rigid cluster decomposition on the hydrogen bond constraints.

Note that the β -sheet is flexible in all 20 structures. We will return to the discussion of β -sheets shortly.

6.4.3 FIRST-ensemble rigidity/flexibility predictions

The output of the FIRST-ensemble Algorithm 6.3.1 on Sso AcP is provided in Figure 6.9. In this ensemble rigidity prediction, helix 2 retains most of its rigidity and

becomes flexible only at its end points. Helix 1, on the other hand, is significantly more flexible. Roughly half of helix 1 is rigid, corresponding to its C-terminal half, and the N-terminal half is flexible. In the flexible half of helix 1, most hydrogen bonds are either broken (i.e. their average strengths do not pass the energy cutoff), or there is not a sufficient number of persistent strong hydrogen bonds over the ensemble, so they are weakened with less bars between hydrogen and acceptor atoms (Algorithm 6.3.1 step 3 (b)). As expected, since the β -sheet was flexible in every NMR model, the β -sheet is also flexible in the ensemble prediction.

There are 53 hydrogen bonds whose average strength over the ensemble met the -1 kcal/mol energy cutoff (i.e. had values < -1 kcal/mol) and were included as constraints. Out of 53 of these hydrogen bonds, 18 had a correction factor applied (using 1 to 4 bars) as they were not persistently strong across the ensemble (i.e. they were present 16 or less times across the ensemble - see FIRST-ensemble algorithm). Out of 18 of these hydrogen bonds, 14 or 78% had energies between -1 kcal/mol and -2 kcal/mol. It is not surprising that majority of the hydrogen bonds that had a bar(s) penalty applied also had slightly weaker energies (close to cutoff). Most of the remaining strong hydrogen bonds (< -2 kcal/mol) received a full 5 bars (edges).

The dilution plot for FIRST-ensemble prediction on Sso AcP is shown in Figure 6.10 (see Chapter 1 and [82, 83, 109, 200] for detailed explanation of dilution plots). Initially, with inclusion of all potential hydrogen bonds (weak and strong) from the entire ensemble, in the FIRST-ensemble prediction of Sso AcP the entire protein is rigid with the exception of the long tail at the N-terminus (top red line). Once most weak hydrogen bonds are broken, it is evident from the dilution plot that β -sheets become flexible and essentially only two significant rigid clusters remain, corresponding to the regions within the two α -helices. As the two rigid clusters correspond to α -helices, the rigid clusters are composed of two contiguous regions in the sequence.

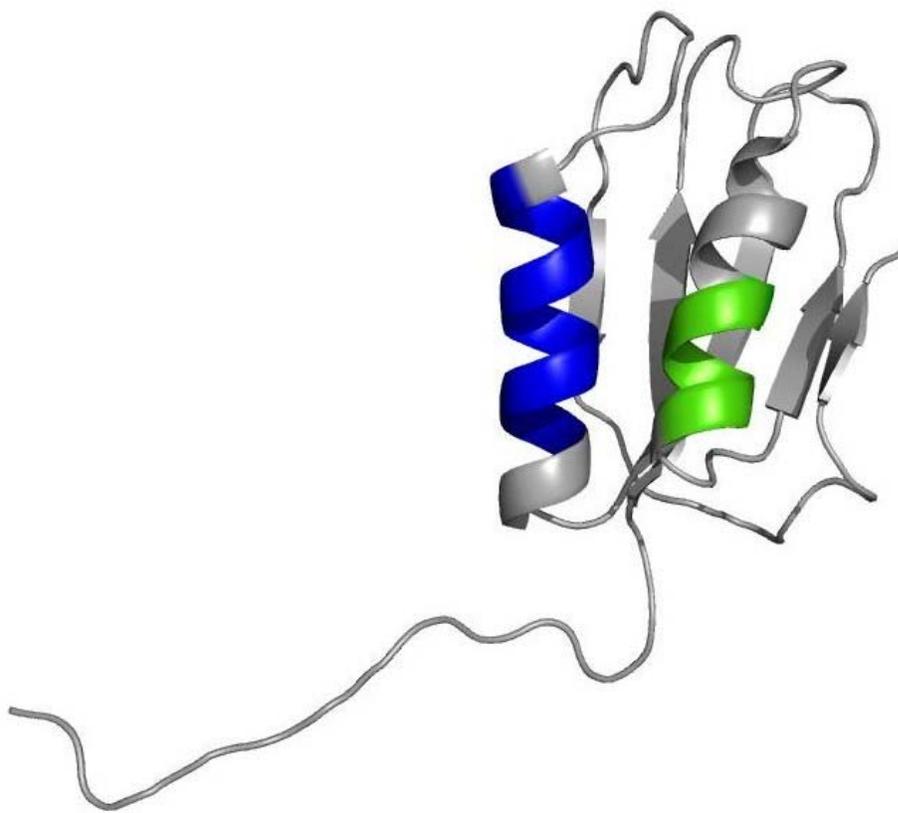


Figure 6.9: Rigid cluster decomposition from FIRST-ensemble, Algorithm 6.3.1. There are two main rigid clusters. Most of α -helix 2 is rigid (blue), with the exception of its end points. On the other hand, α -helix 1 is rigid on its C-terminal half (green), and flexible in the N-terminal half. The β -sheet is flexible.

The fact that the β -sheet in Sso AcP is flexible and the helices are significantly more rigid in the FIRST analysis is concordant with the study by Whiteley [207], which provides precise rigidity-based counting characterization of flexibility and minimum number of constraints needed for simple secondary structure motifs (i.e. rings, loops, α -helices, β -sheets and β -barrels) to become rigid in FIRST. This study reminds us that most isolated perfect β -sheets (i.e. with all backbone hydrogen bonds formed), will generally be flexible in FIRST. With the exception of β -barrels (i.e. a β -sheet with the wrapping set of hydrogen bonds where first and the last strand are connected), an isolated sheet will only become rigid when there are a large number

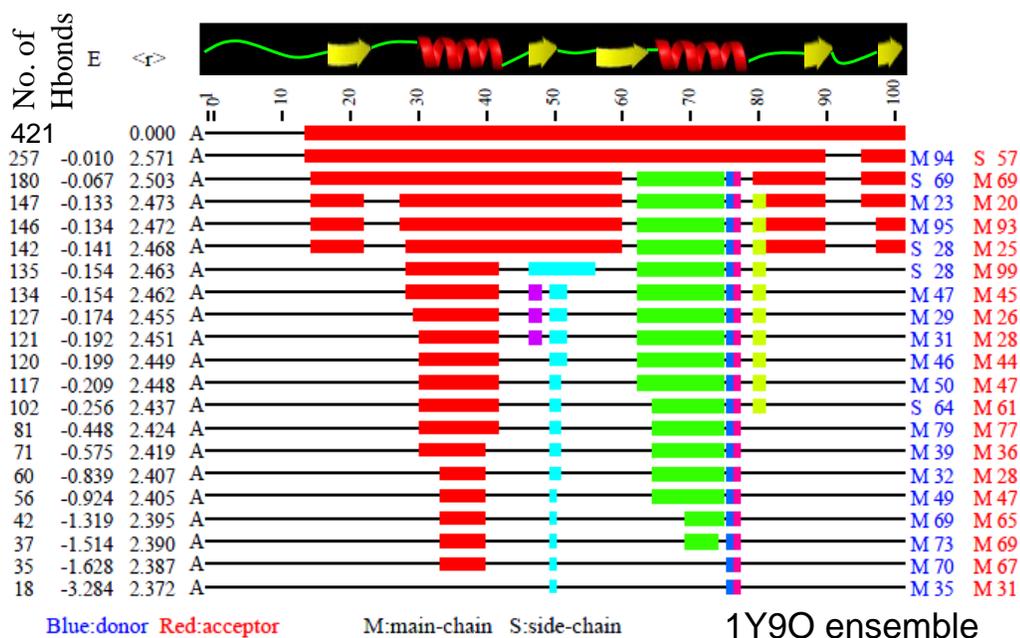


Figure 6.10: Dilution plot of FIRST-ensemble (Algorithm 6.3.1) on Sso AcP. Initially with all the hydrogen bonds included, of which most are very weak (i.e. bad donor acceptor geometry), the entire protein is rigid (i.e. a horizontal block), with the exception of the N-terminus flexible tail. Once weak hydrogen bonds are broken, several rigid clusters form. β -sheet becomes flexible early in the dilution, while the two α -helices retain their rigidity much better. It is suspected that β -sheet becomes flexible, as most side chain strand-strand and strand-helix hydrogen bonds are broken, which are more crucial for the β -sheet rigidity than for the α -helices, which can maintain their rigidity with its internal backbone hydrogen bonding interactions. The columns on the right represent the residue numbers of the donor and acceptor atoms of the broken hydrogen bond at each step.

of strands or fewer strands but with significant lengths (i.e. number of residues). On the other hand, an isolated helix (or part of) will attain its rigidity much easier than a β -sheet (see [207] for details). The fact that we are finding α -helices of Sso AcP to be more rigid than the β -sheets is also consistent with previous FIRST analysis on other proteins. For instance, Wells et al. [200] have observed that as weak hydrogen bonds are broken (diluted), α -helices retain their rigidity much longer than β -sheets.

Even though the β -sheet of Sso AcP is flexible, it has a well-defined structure and a significant number of strong inter-strand hydrogen bonds (see below on accessibility of sheets). In Figure 6.11 we have displayed the β -sheet and the hydrogen bonds that are included as constraints in the FIRST analysis. In particular, we note that most of the hydrogen bonds between the three central strands meet the energy cutoff.⁹ The presence of strong inter-strand hydrogen bonds in the sheet is supported by the anti-parallel arrangement of the four β -strands 1 to 4 (i.e. the N-terminus of one strand is adjacent to the C-terminus of the next strand). In the anti-parallel sheets the hydrogen bonds between NH and CO groups are nearly planar, which is their preferred orientation, which causes the strongest inter-strand stability [8].

Of course, β -sheets can be very stable secondary structure elements. Typically, in FIRST we expect β -sheets to be rigidified with hydrophobic interactions and a sufficient number of side chain hydrogen bonds. This occurs when they are well-packed against other sheets or helices, as is for instance the case in the amyloid-fibril formation via stacking of β -sheets, where there is both strong inter-strand and inter-sheet interactions [105]. In case of Sso AcP, the β -sheet does not have enough of these additional constraints either within the strands or between the sheet and helices to be rigid. Most of these hydrogen bonding interactions have significantly weaker energy (i.e. closer to 0 kcal/mol) strengths and do not meet the FIRST hydrogen bond energy cutoff. It seems that the importance of these side chain hydrogen bonding interactions to rigidity of β -sheets in the context of FIRST has not been explicitly commented or investigated in the FIRST literature.

⁹Recall that the impact of the hydrogen bonds as constraints on the rigidity outcome of FIRST is only dependent on the number of hydrogen bonds that are included. The total number of independent constraints is the only determining factor in FIRST rigidity predictions. So, a hydrogen bond removes same degrees of freedom from the system if it has energy of -1.5 kcal/mol or much stronger energy such as -5 kcal/mol.

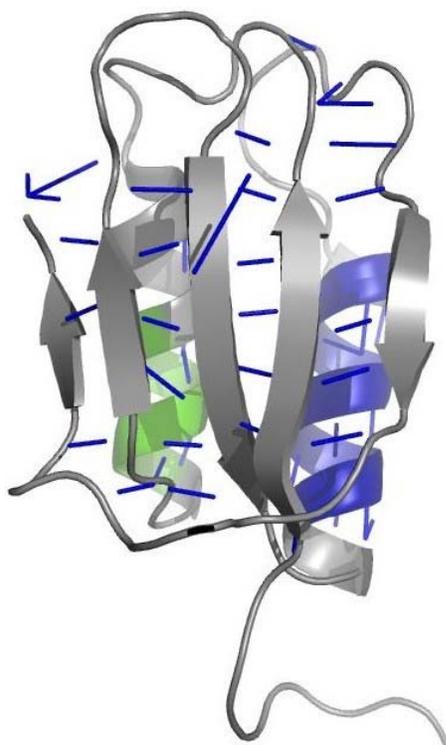


Figure 6.11: Output of FIRST-ensemble, same as in Figure 6.9 with a different orientation. Note the significant number of strong inter-strand hydrogen bonds (indicated with blue lines) in the flexible β -sheet.

The FIRST-ensemble prediction on Sso AcP captures the variations in rigid clusters between individual models. The clear advantage of the FIRST-ensemble prediction over the traditional FIRST runs performed on NMR files using a single snapshot is that FIRST-ensemble incorporates the structural information from all NMR models. By incorporating and averaging the hydrogen bonding strengths and persistence from all structures, FIRST-ensemble is able to capture the variations among FIRST predictions on individual snapshots of the ensemble. Furthermore, in contrast to the previous FIRST study [67], where the authors performed rigidity analysis of individual MD generated snapshots, and then averaged rigidity results (i.e. flexibility index), our ensemble based FIRST prediction is attained using only a

single (adapted) FIRST run. This way we can continue to utilize and take advantage of the fast computational speed of FIRST.

As it was discussed earlier, rigidity and flexibility prediction on its own does not lead to a conclusive measure of HDX (i.e. we also need solvent accessibility). However, it is already evident that the rigidity/flexibility predictions using FIRST-ensemble are much better matched with the experimental HDX data on Sso AcP (Figure 6.7) than the traditional FIRST runs on a single NMR snapshot.

In the FIRST prediction on the most representative model (model 1), both helices are entirely rigid, while in the FIRST-ensemble prediction, helix 2 is mostly rigid with ends becoming flexible, and more importantly, only half of helix 1 is rigid. From the experimental HDX data, we can see that roughly half of helix 1 and most of helix 2 except its ends are protected from HDX (slow exchangers in helices, Figure 6.7). It is certainly not uncommon for parts of the helices to be unstructured and flexible, but considering the hyperthermophile nature of this protein, it is remarkable that FIRST-ensemble is able to capture sub-structural flexibility of helix 1. Clearly, in using only one structure out of the ensemble as an input into FIRST, important structural variations across the ensemble is lost (leading to local hydrogen bonding geometry changes), particularly the conformational variations in the N-terminal half of helix 1. On both α -helices the FIRST-ensemble predicted rigid regions give a very good indication of the regions that will be protected from HDX. As the rest of the protein is flexible, particularly the β -sheet, we cannot yet draw any definite comparisons or conclusions with respect to the HDX data. For this, we turn to the solvent accessibility of Sso AcP.

Remark. We recall that in the FIRST-ensemble Algorithm 6.3.1, we have only used: (a) the average hydrogen bond strengths over the ensemble (step 2 of algorithm) and (b) a correction function which incorporates the persistence of strong hydrogen

bonds by assigning a correct number of bars (1 to 5) between hydrogen and acceptor atoms (step 3 of algorithm).

We had claimed that hydrophobic interactions would not need to be averaged out over the ensemble. In order to test out this claim, we ran the FIRST-ensemble algorithm with averaged and corrected hydrogen bonding input file on all the individual NMR models. What we found is that an identical FIRST-ensemble rigid cluster decomposition is obtained in all 20 models, and in fact virtually indistinguishable dilution plots (not shown here). Once the averaged hydrogen bonding file is created, the FIRST-ensemble is performed only once on a single snapshot, and which model (snapshot) we choose has no impact on the results. As we are running the FIRST-ensemble with the same set of averaged hydrogen bonds, the only potential difference from model to model in the created constrained graph can be due to the differences in the hydrophobic interactions (i.e. due to conformational changes between models, some different hydrophobic interactions could be identified). Our findings on Sso AcP show that choosing a different set of hydrophobic constraints has no impact on the rigidity results. This also suggests that the sensitivities of running regular FIRST analysis on individual NMR models on rigid cluster decomposition was caused solely by changes in the identified hydrogen bonds and not hydrophobic contacts. These findings on the NMR conformers are in agreement with a previous FIRST/MD study [122]. The authors also report that hydrophobic tethers have less impact on rigidity than hydrogen bonds, and when they are switching partners in the MD simulation, it had little influence on the protein rigidity. ■

6.4.4 Solvent Accessibility ensemble data

The plot of solvent accessibility (RSA) for the backbone amide Nitrogens of Sso Acp, averaged over the ensemble, is given in Figure 6.12, using the calculations from the

WHATIF server. In Figure 6.13 we have coloured the structure with the solvent accessibility colouring scheme as described in section 6.4.4.

The first thing that stands out is that the three β -strands (strands 1, 2 and 3) are almost completely buried (coloured in blue). In fact, almost all residues in these three strands have 0% solvent accessibility. Having no backbone amide (NH) accessibility over the ensemble suggests that these strands should be well protected from undergoing HDX. Indeed, the experimental HDX evidence tells us that the three central β -strands (blue residues in the three strands, see Figure 6.7) correspond to the regions of the protein that are the most protected residues from undergoing HD-exchange (i.e. the slowest exchangers). The two side strands are more solvent accessible. β -strand 4, located behind α -helix 2 has both buried and almost completely exposed amides. At the C-terminus of strand 4, backbone amide Nitrogens of residue phenylalanine87 is almost completely exposed (47 % accessibility) as it is directed towards the solvent and is bound by two completely buried (0 % accessible) serine86 and serine88, whose amide protons are pointing towards the Sso AcP structural core. This is in good correspondence with the experimental HDX, where a ‘Very Fast’ phenylalanine87 exchanger is bounded by two ‘Very Slow’ serine exchangers (Figure 6.7). β -strand 5, the short strand located at the opposite edge of the β -sheet appears to be overall more solvent accessible. This again is in reasonable agreement with the experimental HDX.

We had indicated earlier that solvent accessibility data will not be used on α -helices, as it should not be a reliable indicator of HDX due to the general *amphipathic* nature of many helices (i.e. one side facing the core, and other facing the solvent). As anticipated, we find that the helices of Sso AcP have mixed solvent accessibility measures, with one side generally solvent accessible, and the other side being more

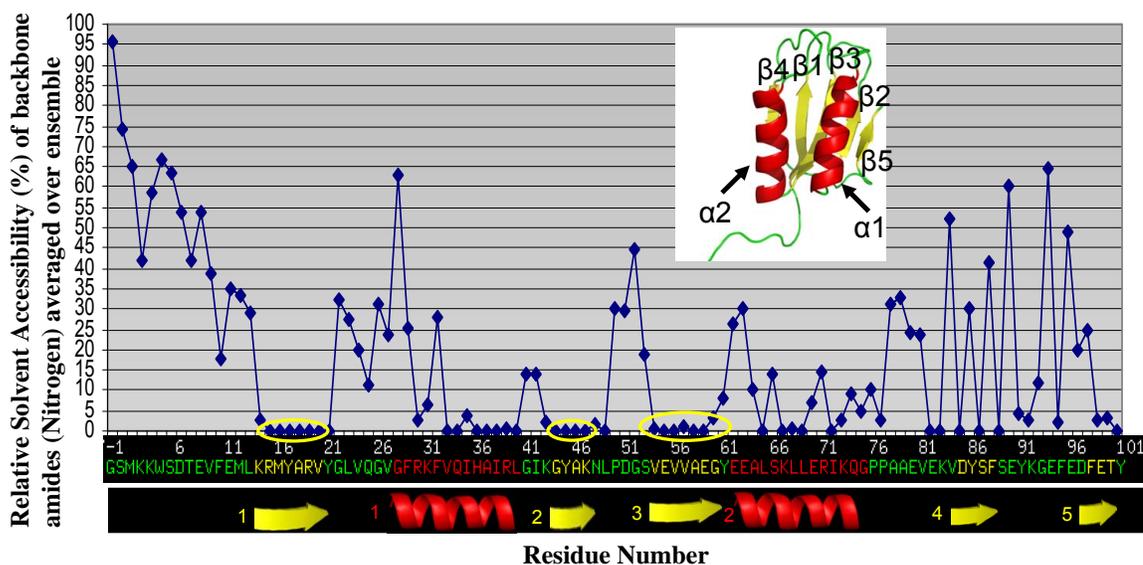


Figure 6.12: Plot of solvent accessibility (RSA) of Sso AcP of backbone amide (NH) averaged over the ensemble, obtained from WHATIF [195]. RSA values are shown on the vertical axis, and amino acid sequence (main-chain) with secondary structures on the horizontal axis. Note the three central β -strands (strand 1, 2, and 3, circled in yellow) are almost completely buried.

solvent inaccessible. This becomes more apparent if we look at the side view orientation of the protein (Figure 6.13 (b)). Here we observe that the two sides of α -helices that are facing one another or towards the β -sheet are mostly buried, and the opposite side generally more solvent exposed. In the N-terminal half of helix 1, several residues (backbone amide (NH)) have very low accessibility (residues Arginine30 and Valine33 are almost completely buried, Lysine31 is only somewhat exposed). However, we have already seen that this flexible half of helix 1 is very unstructured, and from experimental HDX evidence it does not represent a slow exchanging part of the protein. Similarly, solvent accessibility data on helix 2 would not be a good predictor of HDX, and is not in good agreement with the experimental HDX profile.

In addition to the amphipathic nature of helices, we can envision another reason why solvent accessibility on α -helices is not a sufficient measure to probe

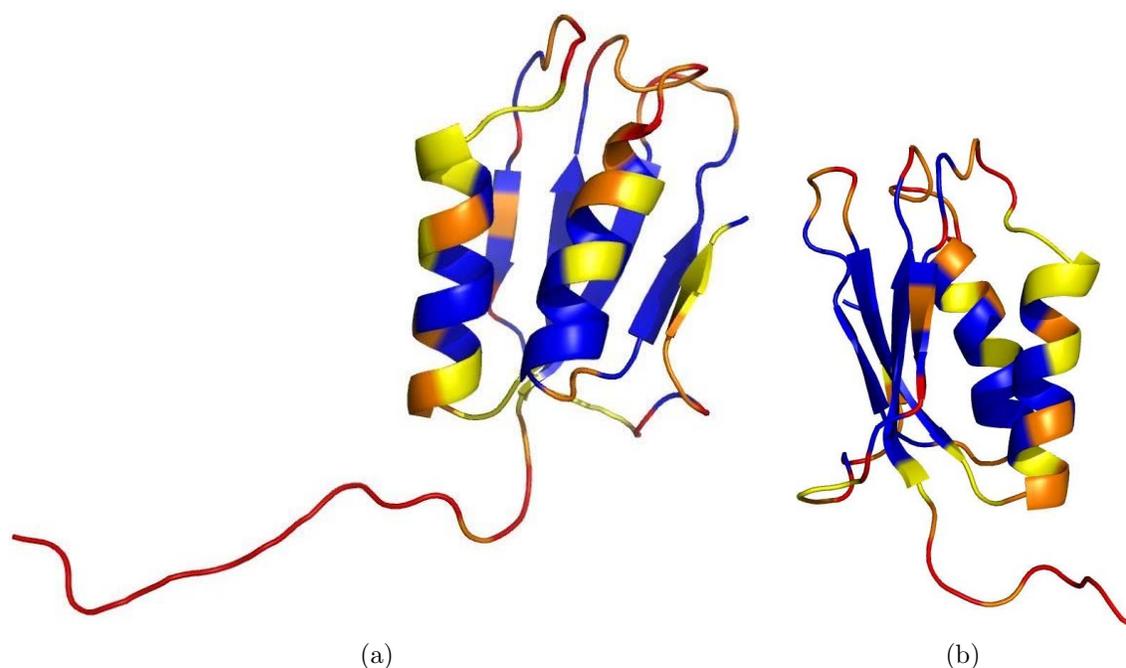


Figure 6.13: Solvent accessibility (RSA) of Sso AcP of backbone amide (NH) averaged over the ensemble. The blue regions are almost completely buried (solvent inaccessible) ($\text{RSA} \leq 4\%$), followed by yellow regions - somewhat exposed, orange - mostly exposed, and red - almost completely exposed (a). Same display with different orientation (b) which indicates the mixed accessibility of α -helices, with the side pointing towards the interior of the protein being generally more buried.

HDX. Unlike β -sheets, which can be declared flexible by FIRST and yet have a well-defined structure with strong inter-strand hydrogen bonding network, substructural flexibility in an α -helical structure (as in helix 1 in Sso AcP) is a direct consequence of missing (weak) local helical hydrogen bonding interactions (see previous discussion or [207]). In the flexible part of the α -helix, its backbone amides are not involved in hydrogen bonding for the maintenance of secondary structure. So, even if part of the flexible helix is computationally predicted to be buried on a snapshot(s), this region of a helix can still potentially undergo backbone (nontrivial) motions and turn a previously buried backbone amide proton into a solvent exposed amide proton, hence still allowing exchange to occur.

The overall solvent accessibility that we found on Sso AcP is in good agreement with the general patterns found in [119, 153]. Lins et al. have performed an in-depth analysis of solvent accessibility on a nonredundant bank containing roughly 600 proteins. They have observed that residues from random coils/loops, and outside β -strands are usually the most solvent accessible regions in the protein. α -helices have a mixed solvent accessibility/buriedness distribution, segregating the hydrophobic and hydrophilic residues, confirming the general amphipathic character of the helical fold. In [153] mixed/intermediate accessibility is also reported on number of α -helices. β -sheets (both parallel and anti-parallel), particularly the central β -strands, are consistently the most solvent inaccessible regions in the protein, as is the case for our case protein Sso AcP.

In summary, solvent accessibility on Sso AcP gives a good estimate of HDX on the β -sheet, and a poorer prediction on α -helices as we had expected. The remaining flexible regions corresponding to loops and the long flexible N-terminal tail are mostly solvent accessible, and overall correspond well with the experimental determined fast HD-exchangers. It is well known and reported on many proteins that flexible and highly mobile regions in the protein are often solvent accessible [218]. As we are calculating the average solvent accessibility over the entire ensemble, any highly flexible region that retains its solvent accessibility over the entire ensemble has a higher likelihood to remain solvent accessible. However, we should remind ourselves that unlike structured β -sheets, in general any highly flexible region (such as the tail and loop regions in Sso AcP) is extremely unstructured with a few stable hydrogen bonds. So, the underlying message is that solvent accessibility data needs to be considered with care.

6.4.5 Combined rigidity and solvent accessibility predictions of HDX

We have seen how both rigidity (FIRST-ensemble) predictions and solvent accessibility measures are valuable tools that can be used to computationally probe HDX. However, on their own, rigidity or solvent accessibility would lead to an inefficient overall predictor. Rigidity approach may not be well suited on β -sheets, as sheets can often come up flexible with FIRST analysis (see above discussion). This is the case with Sso AcP, where the backbone amides of β -sheet are inaccessible to the solvent with a well structured and strong inter-strand hydrogen bonding interactions. On the other hand, rigidity is a better predictor on α -helices. By incorporating both analysis, an improved prediction of HDX can be achieved.

The combined FIRST-ensemble rigid clusters and solvent accessibility on Sso AcP is shown in Figure 6.14 (a) (see Section 6.3.4 and Algorithm 6.3.2 for details). We recall, the blue region in the α -helices corresponds to the two rigid clusters found by FIRST-ensemble algorithm (Figure 6.9). The remaining blue regions, notably the three central β -strands, and some residues in β -strand 1 represent the ‘solvent inaccessible’ regions. These (combined) blue regions are the predicted regions in the protein that will most likely be protected from undergoing exchange (i.e. slowest exchangers). Comparing this computational prediction with the experimentally obtained HDX profile of Sso AcP, we see that the the predicted HDX protected regions are in very good agreement with the experimentally determined very slow exchangers. These results confirm that the most accurate results are attained by combining both rigidity and solvent accessibility.

As it can be difficult to visualize these results on the 3-dimensional representation of the protein, in Figure 6.15 we have expressed the computationally determined regions that are most likely to be protected from exchange (i.e. slow exchangers) on

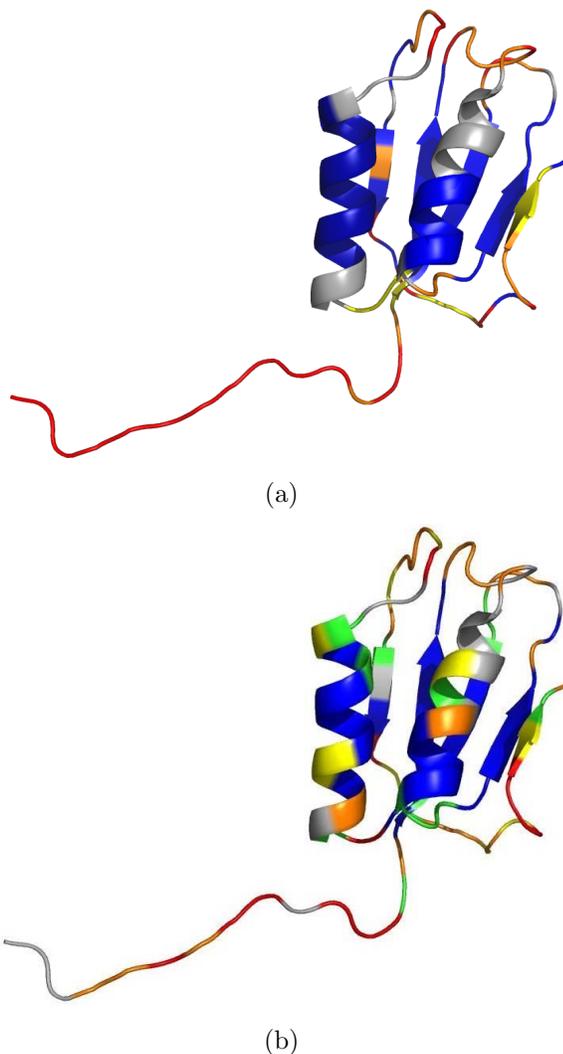


Figure 6.14: Combined rigidity (FIRST-ensemble) and solvent accessibility on Sso AcP, see Algorithm 6.3.2 (a), and experimentally derived HDX profile of Sso AcP (b). The blue regions (computationally predicted to not undergo HDX) in (a) (obtained by combining the rigid clusters and buried residues) match exceptionally well with the experimentally determined slow exchangers. The remaining regions are also in a good agreement.

the 1-dimensional (backbone) representation of the protein, along with the experimentally determined slow exchangers.

In line 1 the blue regions represent the experimentally determined very slow exchangers (most protected residues). For further comparison, we have also shown

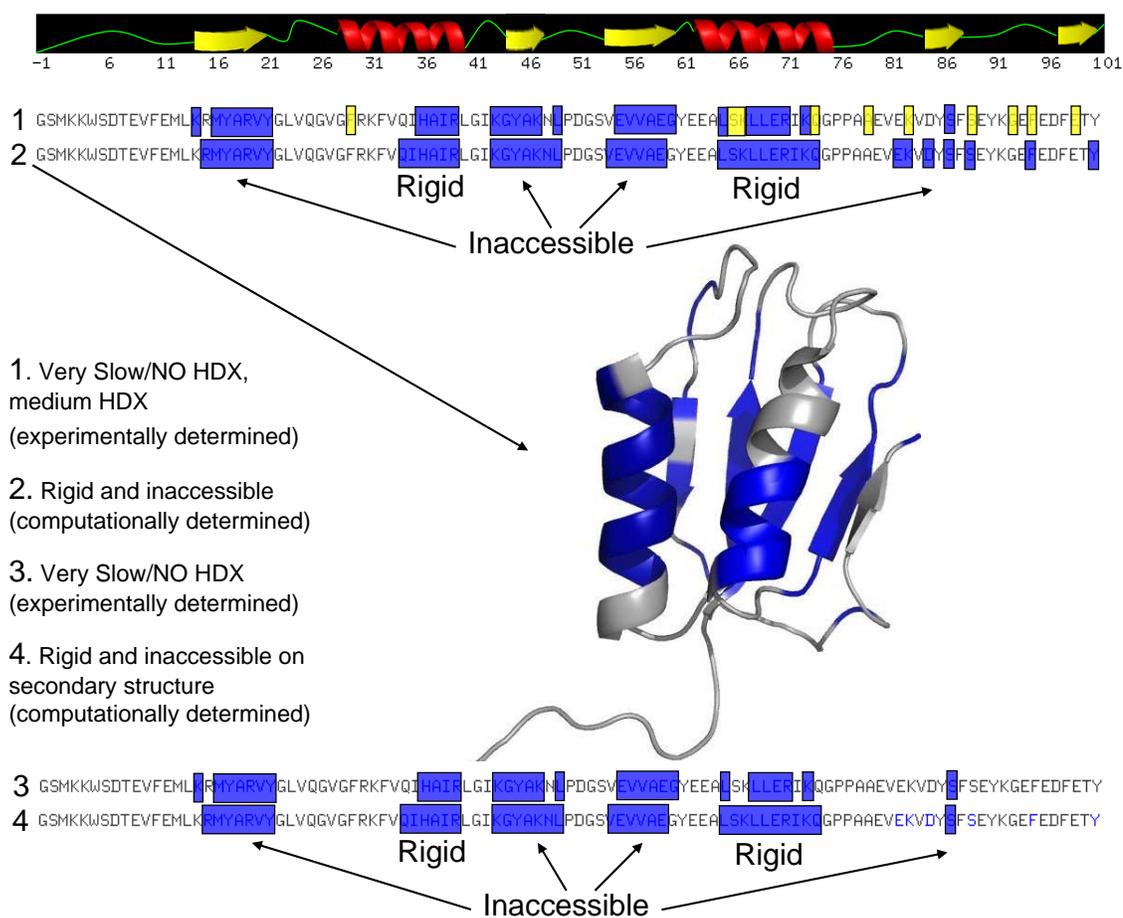


Figure 6.15: Comparing the computationally determined regions that are least likely to undergo exchange with the experimentally determined regions that rarely undergo exchange (i.e. slow exchangers) on the 1-dimensional (backbone) protein chain. In 1. blue regions represent the experimentally determined slowest exchangers and yellow represent the medium exchangers. In 2. blue regions are combined FIRST-ensemble rigid regions with inaccessible regions. In 3. experimentally determined slowest exchangers are shown only on secondary structures, and in 4 computational determined slow exchangers are shown only on secondary structure. The computationally predicted HDX protected residues (particularly on the secondary structural elements) and the experimentally observed protected residues, are in close correspondence and show a significant overall overlap.

the medium exchangers residues in yellow, which are the next slowest exchangers, and can still be considered to have sufficient protection from exchange [211]. The medium exchangers are mostly scattered (isolated) in the very unstructured and flexible loop regions in the C-terminal part of the protein. In line 2 we have depicted the

computational determined slow exchangers (i.e. rigid and almost completely buried residues), with non-coloured regions being the predicted regions that are not protected from undergoing exchange. There is significant overlap between the computational predictions and experimentally determined results. Even in the highly unstructured loop regions, the buried (inaccessible) residues are somewhat matched with medium exchangers.

In lines 3 and 4 we have compared the predictions only on the secondary structures. Line 3 corresponds to the experimentally determined very slow exchangers on secondary structures, and line 4 corresponds to the computationally determined most protected regions (slowest exchangers) on secondary regions. We observe that the computational predictions are remarkably well matched with experimentally determined slowest exchangers on all the major secondary structural regions.

In summary, our computational approach gives a very good prediction of the regions that are the most protected from undergoing exchange as supported by experimental data. We have also observed a good correlation on the the loops and the N-terminus tail, but as expected these regions are highly flexible and unstructured with a few stable hydrogen bonds, and are much harder to probe than the secondary structures. Our findings on Sso AcP support our initial hypothesis that combined rigidity and solvent accessibility predictions can be used to computationally probe HDX.

Remark. For completeness, we also ran the FIRST analysis on the crystal structure (i.e. single snapshot) of Sso AcP (pdb: 2bje). We found that FIRST identifies all the secondary structures (both helices and the beta sheet, including side strands) to be entirely rigid. Given this overprediction of rigidity, FIRST applied on the crystal structure did not provide us with valuable information and ability to predict HDX. Since HDX is an ensemble experiment, it is advisable to use NMR ensemble (models)

with the FIRST-ensemble algorithm (combined with solvent accessibility ensemble data) when predicting HDX. ■

6.5 Concluding remarks and future work

The aim of this chapter was twofold: to introduce a novel algorithmic way of computing rigidity of ensembles adopting FIRST methods and to show that combining rigidity and solvent accessibility can lead to good fast computational predictions of HDX, as it was applied on hyperthermophile protein Sso AcP.

With regards to the first algorithmic point, we found that there is a significant variation in the rigid cluster decompositions across individual NMR models. FIRST-ensemble prediction incorporates the structural information and variations from all the models and we have shown that the ensemble rigid cluster decomposition are better matched with experimental HDX on Sso AcP.

As part of future work, it would be valuable to apply our FIRST-ensemble algorithm on a larger class of NMR proteins, where it would be desired to have experimental data (such as the high coverage HDX profile on Sso AcP) for comparison and further validation. In other current work [155], using rigidity adapted coarse graining MD simulations [192], initial promising results suggest that FIRST-ensemble analysis on NMR file of hemagglutinin fusion-peptide is a better (appropriate) choice for rigidity prediction than the single snapshot FIRST analysis.

From our findings on Sso AcP, and general analysis on FIRST in [200], we suggest that the FIRST rigidity analysis on NMR ensembles should be based on the FIRST-ensemble predictions to make them more robust. Since the methods and algorithms we have introduced can be generalized to a broad range of ensemble data, whenever structural information of a protein is provided in an ensemble setting (NMR models, MD snapshots, etc.), we invite the current and future users of FIRST to use

the entire conformational ensemble information and consider utilizing the suggested techniques and methods given in this chapter.

We can foresee further applications using our FIRST-ensemble protein rigidity/flexibility predictions. For instance, rigid regions in the ensemble prediction that have many persistently strong hydrogen bonds (i.e. no bar penalty had to be applied) could be called persistently-rigid (or super rigid) over the ensemble. One could then visually denote (i.e. in Pymol) these rigid clusters (after they are normalized with respect to how many atoms (vertices) the rigid cluster has) with a different colouring scheme. On the other hand, rigid regions that have a significant number of hydrogen bonds that do not persist in all models (i.e. have bar penalties applied) could be called marginally (or weakly) rigid over the ensemble and coloured appropriately. This should give a more precise and deeper meaning of the rigid cluster decompositions when ensemble data is available (rather than just rigid or not representation). This should be further explored in extending and refining the FIRST software to ensembles.

Both rigidity and solvent accessibility are important tools in probing HDX, but best prediction is achieved when both of these measures are combined. To test out this hypothesis, we have shown that our combined ensemble rigidity/accessibility algorithm predictions on Sso AcP is well matched with the experimental HDX profile of Sso AcP (which is an ensemble measurement). In particular, we have shown that our computational predictions of regions that are protected from HDX and those that will likely undergo fast exchange (i.e. not protected from HDX) are in very good agreement with the experimental HDX data. The clear advantage of these techniques is that they offer very fast computational methods in probing an expensive and laborious experimental method. As part of ongoing future work, the real power and usefulness of the algorithms and methods developed in this work will be further

strengthened with testing on additional proteins (with a solved NMR structure) with the availability of comparable (ideally high coverage native-state) experimental HDX data.

The usefulness of FIRST flexibility analysis lies in its simplicity and the fast computational predictions, as it was demonstrated in the the current and previous two chapters and in numerous previous FIRST studies. The methods and techniques developed in this chapter should further enhance the capability of FIRST and offer new tools and research avenues in predicting rigidity of ensembles and in computational predictions of HDX.

We now switch themes from rigidity and algorithmic applications to proteins that were the focus of Chapters 4 to 6, and turn to an application of rigidity techniques and the pebble game algorithm to problems related to decompositions of graphs and mechanical linkages in mechanical engineering.

Chapter 7

Decompositions of Linkages, Assur graphs and the pinned pebble game algorithm

7.1 Overview

The work in this chapter is joint work with Offer Shai and Walter Whiteley and most of it appears in [166, 174].

The decomposition of a linkage into fundamental minimal components is a central tool of analysis and synthesis of linkages. The concept of Assur graphs (previously named Assur groups in the Engineering literature) was originally developed by Leonid Assur [7] in 1914 and is widely used in the kinematical community to decompose linkages into minimal pieces whose components could be merged to give a simplified overall analysis [128, 138, 162].

In this chapter we prove that every pinned d -isostatic (minimally rigid) graph (grounded linkage) in dimension d has a unique decomposition into minimal strongly

connected components (in the sense of directed graphs), or equivalently into minimal pinned isostatic graphs, which we call d -Assur graphs (see below for definitions). We also study key properties of motions induced by removing an edge in a d -Assur graph - defining a stronger sub-class of strongly d -Assur graphs by the property that all inner (non-pinned) vertices go into motion, for each removed edge. The strongly 3-Assur graphs are the central building blocks for kinematic linkages in 3-space and the 3-Assur graphs are components in the analysis of built linkages. The d -Assur graphs share a number of key combinatorial and geometric properties with the 2-Assur graphs, including an associated lower block-triangular decomposition of the pinned rigidity matrix.

We will highlight some problems in combinatorial rigidity in higher dimensions ($d \geq 3$) (i.e. the lack of Laman-type of theorem) which cause a distinction between d -Assur and strongly d -Assur which did not occur in the plane. This work is the first time decompositions of Assur graphs in general dimension d is introduced. All the work in this chapter is on the pinned bar and joint structure. We will make some concluding remarks about the core techniques and algorithms that easily generalize and can be adapted (with stronger results) to body-bar, body-hinge types of kinematic structures.

We also introduce a 2-dimensional pinned pebble game algorithm as an efficient algorithm for studying rigidity of linkages, and when combined with strongly connected decompositions, we attain a new algorithm in mechanical engineering for decomposing 2-dimensional linkage into 2-Assur components. The decomposition algorithm also extends to frameworks in higher dimensions with the assumption that the pinned framework is isostatic.

We also demonstrate how extensions of the pebble game algorithm on a pinned 2-isostatic graph (minimally rigid in dimension 2) can be used to determine which

vertices will become mobile, and which vertices will remain rigid, when an edge is removed. This has important practical applications to linkages in mechanical engineering.

7.2 Introduction

Closely related to rigidity theory is the theory of planar and spatial linkages [70]. A planar or spatial linkage, is basically a (flexible) bar and joint framework (see Chapter 2) where some joints (vertices) are held fixed (“pinned down”) to the plane (or 3-space), which we will call a *pinned framework* (see Figure 7.1 for an example of a 1 DOF linkage). When a vertex (joint) is *pinned* it cannot move, but the edges (bars) at that vertex can freely rotate about that vertex. The idea is to pin some vertices so that the trivial rigid body motions are removed. Unpinned vertices will be called *inner vertices* (precise definitions are given in Section 7.4.1.)

One historical interest in linkages stems from the types of curves that they can trace out (i.e. put a pen on one inner joint, and observe the curve that is traced out as the linkage moves in the plane). For example, a simple linkage in the plane that consists of a single bar pinned at the end has 1 DOF, can be used to trace out a circle. One of the most famous early problems in theory of linkages is to find a planar linkage that can trace out a straight line. In 1784 James Watt found a simple linkage that provided a good approximation to straight line motion, and this linkage is still used today, for instance in automobile suspensions. Engineer M. Peaucellier, in 1864 found the exact solution [70], with the Peaucellier linkage. The early history of this problem and theory of linkages is provided in a published lecture by A.B. Kempe in 1876 [103].

Today linkages (also generally referred as mechanisms or kinematic chains) are studied in wide areas and have numerous applications in robotics, Computer Aided

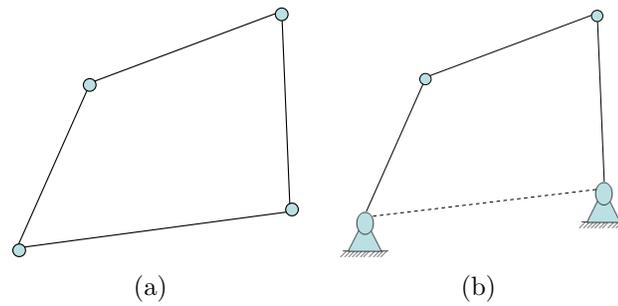


Figure 7.1: A flexible bar and joint framework in the plane with 4 total DOF (3 trivial + 1 internal DOF) (a). If we pin down (i.e. ground) two vertices (obtaining pinned vertices – joints) of the ends of the bar (indicated by a small hatched triangle), we get a pinned bar and joint framework with 1 DOF. Only the two inner vertices are mobile, the pinned joints do not move. In planar mechanism terminology, this simple and very common linkage is called a *four-bar linkage* (it is imagined that the fourth bar between the two pinned vertices is present). This linkage is said to be a 1 DOF linkage.

Design (CAD), engineering and manufacturing [136, 162, 179]. Linkages also occur in art. One of the fascinating linkages is the the Jansen’s linkage, designed by a Dutch kinetic sculptor Theo Jansen (<http://www.strandbeest.com>), who has been building marvellous wind propelled walking kinetic sculptures (walking machines - Figure 7.2).



Figure 7.2: A Theo-Jansen kinetic sculpture can be viewed as a large (floating) linkage (source credit: <http://www.strandbeest.com>).

Our work is motivated by the mechanical engineering applications, specifically in decompositions of linkages.

The decomposition of any system of constraints into small basic components is an important tool of design and analysis. In particular, the decomposition of a mechanical engineering linkage into minimal mechanical components is a central tool of analysis and synthesis of mechanisms [7, 128, 163, 165, 174]. Figure 7.3 illustrates the analysis: the initial mechanical engineering plane linkage (Figure 7.3(a)), is transformed to a flexible pinned framework (Figure 7.3(b)); designating one of the links ¹ to be a *driver* ², then adding an extra bar, or pinning the end of the driver to the ground ³, this becomes an isostatic (minimally rigid) ⁴ pinned framework (Figure 7.3 (c,d)).

The focus of this chapter is the study of specific decompositions, of such associated pinned isostatic frameworks in dimensions 2 and 3. The individual components of the decomposed pinned isostatic framework that we will look at are called Assur graphs (which will be discussed shortly) ⁵. However, for completeness, the theorems and definitions will be generalized whenever possible to all dimensions $d \geq 2$. We

¹In a pinned bar and joint framework (linkage), links are bars (edges) connecting pairs of vertices (joints) in a linkage.

²A driver or driving link is a special type of a link which can be altered (e.g. by changing its length, so the link (bar) does not have a fixed distance). We can think of a driving link as driving or changing the distance between its endpoints as a piston, and in effect causing some (or all) inner vertices to go in motion.

³Note in Figure 7.3, the step from (b) to (d) takes an inner vertex and shifts it into a pinned vertex (i.e. pin the inner vertex of a driver) creating a pinned isostatic framework in the plane. This is a standard mechanical engineering practice [166]. This can be thought of as a two step process passing through (c). We add the bar to block the motion of the inner end of the driver. Then we pin the inner end of the driver. We can also apply a reverse operation and ‘release’ a pinned vertex into a new inner vertex, and attach the inner vertex with 2 edges to the ground (d) to (c).

⁴Recall that in rigidity theory, isostatic is synonymous with minimal rigid. In engineering the term that describes minimal rigidity is *statically determinate* [174].

⁵Assur graph are known as Assur groups in mechanical engineering in the sense that they are built from a collection (or group) of a specified set of links [163].

emphasize that our decomposition goes further into the framework than the decomposition of a flexible framework such as (b) into rigid components. Through this decomposition, we can address additional questions.

For plane linkages, this decomposition has been an important tool in the mechanical engineering literature over some decades [7, 128]. Our central goal is to extend the previous decomposition for plane pinned isostatic frameworks [163] to pinned isostatic frameworks in d -space [174]. In developing this extension, we present some key additional properties of the Assur decomposition in the plane (and in d -space) in terms of lower block-triangular decompositions of the associated pinned rigidity matrix (section 7.4). In the analysis, we also draw some new connections to the theory of strongly connected decompositions of directed graphs and their associated algorithms.

The basic motivation and interest in Assur decompositions by mechanical engineers is that it simplifies analysis of various properties of the the pinned framework. The general focus on decompositions of a pinned isostatic framework (constraints) down into smaller fundamental components (Assur graphs) which cannot be further divided is the central problem in analysis and synthesis of mechanical linkages [128, 133, 162]. Instead of performing the complex geometric and algebraic analysis (such as analysis of the possible configurations of mobile joints, analysis of dead-end positions where some vertices jam up [138, 165], etc.) of the entire framework at once, mechanical engineers would like to obtain the Assur decomposition and analyze the (smaller) individual Assur components one by one. This is one effective way that is used by mechanical engineers to study the overall motion of the framework, building back up one component at the time (i.e. synthesis). The practice of decompositions is also common in other fields, where the large system is decomposed into irreducible

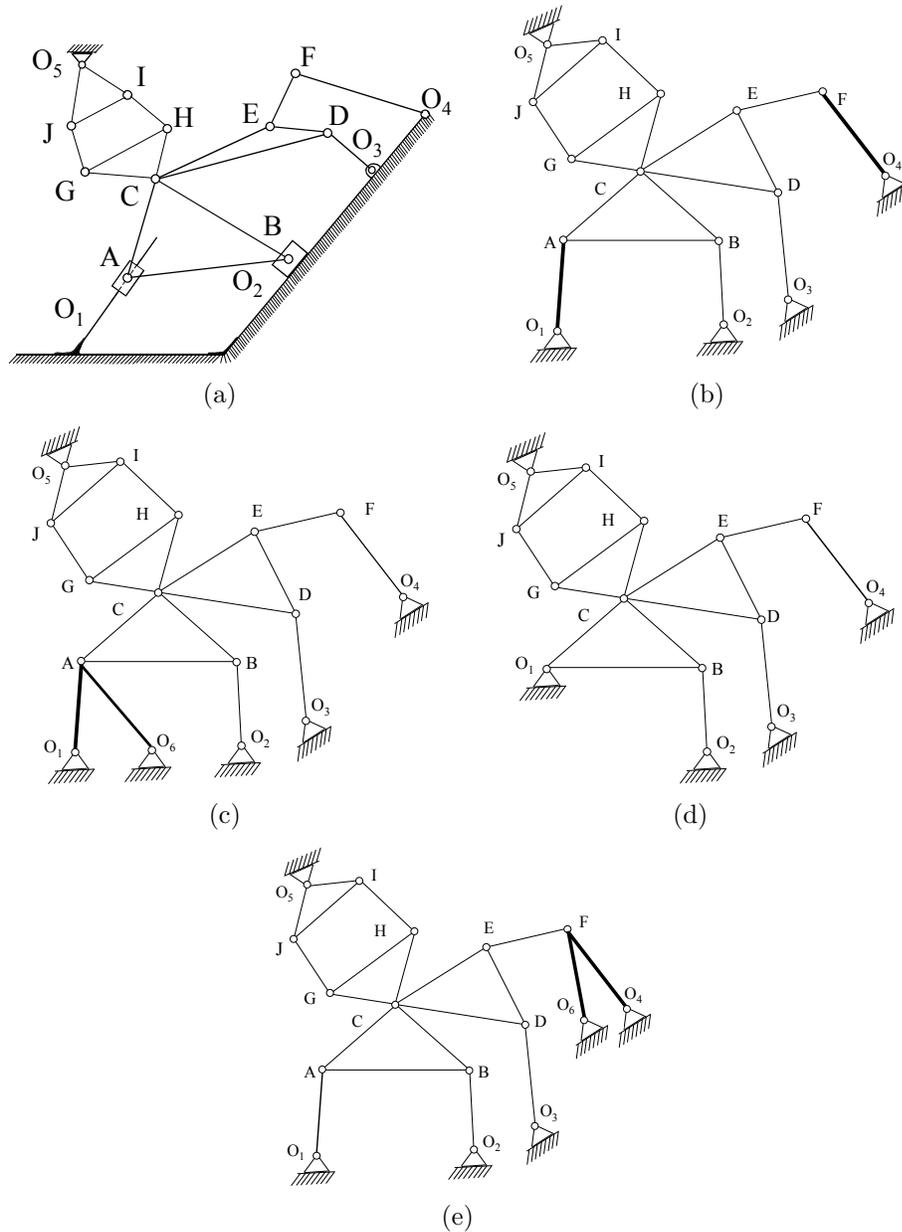


Figure 7.3: A mechanical engineering plane linkage (a) is translated into a flexible pinned framework (b). Assuming the chosen driver is the link from A to the ground, with one added bar (c) or by pinning the end of the driver to the ground (d) we get an isostatic pinned framework. If we specify the link from F to the ground as a designated driver, with one added bar, this results in an alternate pinned isostatic framework (e). Translation from (a) to (b) takes pinned (inner) joints to pinned (inner) vertices, links to bars (edges). The slider joint (for instance at B) becomes an inner vertex with an extra bar attached to the ground, which is a standard mechanical engineering practice and preserves the same DOF count of the linkage [133, 166].

components, such as in CAD systems [136], where the individual components are studied separately, and then re-combined.

Section 7.3 highlights a unique decomposition of directed graphs, using only properties of the graphs of the constraints captured in directions on the edges. A key first step is generating a directed graph for the constraints, which will be directed towards the ‘ground’ of the linkage. This directed graph is then decomposed into strongly connected components (components in directed cycles in the directed graph), using a standard combinatorial result which can be implemented using various algorithms, such as Tarjan’s Algorithm [182]. Overall, the strongly connected decomposition is presented as an acyclic graph with condensed nodes for the strongly connected components (Figure 7.4). In this decomposition, the strongly connected components can be recognized visually as separated in the original directed graph by directed cut-sets.

As a useful invariant property of this decomposition, we will show that two equivalent orientations of a given (multi)-graph (orientations with an assigned out-degree for each vertex) will produce the same strongly connected components, in the strongly connected decomposition. This is important to techniques in section 7.4 and to the pebble game algorithm in section 7.6, which include some choices of orientations for edges, where we conclude that these choices do not alter the decomposition.

In section 7.4, we show that, for an appropriate orientation (a d -directed orientation) the strongly connected graph decomposition gives a decomposition of a pinned isostatic graph which coincides with several other key decompositions of a pinned isostatic graph: (i) with a block triangular decomposition of the pinned rigidity matrix (section 7.4.2) and (ii) an associated decomposition into minimal pinned isostatic graphs (section 7.4.4). For pinned linkages in the plane, this shared decomposition coincides with the 2-Assur decomposition in [163].

Here we extend these decompositions to all higher dimensions $d \geq 2$. The minimal pinned isostatic graphs in this decomposition are called d -Assur, and coincide with the plane 2-Assur components of the previous work [163]. This connection to the pinned rigidity matrix also provides a way to generate a directed graph from the original undirected graph, so that we can apply the strongly directed decomposition of section 7.3, if directions were not already supplied by other analysis. This explicit connection to lower triangular matrices is new in the plane, as well. However, in practice we will use the pinned pebble game algorithm as a fast and efficient way to generate the directed graph (section 7.6).

In Section 7.5, we consider extensions to dimension d of a further key property of 2-Assur graphs: removal of any single edge, at a generic configuration, gives a non-trivial motion at all inner (unpinned) vertices. This property represents the mechanically desirable property that a driver replacing this edge causes all parts of the mechanism to be in motion. While this property follows for minimal pinned isostatic graphs in the plane, it is a stronger property in higher dimensions. We use this added property to define the *strongly* d -Assur graphs in all dimensions, as a restricted subclass of d -Assur graphs. This distinction between d -Assur graphs and strongly d -Assur graphs adds another view on the complexity of obtaining a combinatorial characterization (i.e. Laman type of theorem) of generic rigidity in 3-space or higher. We will offer very important examples which illustrate this distinction.

In section 7.6, we introduce a 2-dimensional pinned pebble game algorithm, which determines if the pinned plane framework (linkage) is (generically) isostatic (minimally rigid), if it has redundance and any additional remaining DOF. We use the 2-dimensional pinned pebble game algorithm (and in some special cases in higher dimensions) together with strongly connected decompositions and introduce an efficient algorithm for decomposing 2-dimensional linkages into 2-Assur components.

The pinned pebble game algorithm will be applied on sample linkages, and 2-Assur decompositions will be obtained. We will also provide an overview of the key properties and advantages of the pinned pebble game algorithm, extending the discussions in Chapter 3.

Building on the work in Chapter 3 and our extensions of the pebble game, we show that upon removal of any edge from a pinned 2-isostatic graph, and by tracking the possible movement (distribution) of the free pebble in the directed graph, we can determine which vertices will be in motion, and which remain rigidly attached to the ground. This answers important practical problems in linkages such as insertion (or replacement) of a driver, and its effect on the mobility of inner vertices. This is the first time the pebble game algorithm is used to analyze and answer important questions related to analysis and synthesis of linkages in the mechanical engineering community.

We also highlight some key unsolved problems in the rigidity theory for frameworks in dimensions higher than 2 which this analysis brings back into focus in terms of pinned structures.

In the concluding section we mention some further directions and extension of these techniques and decompositions to the alternate body-bar/body-hinge frameworks, which have good combinatorial rigidity theories in dimension 3 and higher (see Chapter 2), with d -Assur again coinciding with strongly d -Assur, and good fast pebble game algorithms (see Chapter 3) for these structures.

7.3 Decomposition of Pinned Directed Graphs

As we have seen in Chapter 2, in combinatorial rigidity theory, the rigidity of a given framework is a property of an underlying undirected graph. However, directed graphs

come up in several important applications and algorithms in rigidity theory. For example, in the control theory of formations of autonomous agents, the constraints on distances between agents are represented as a directed graph [43, 80]. The output of the fast pebble game algorithm [114, 172, 174] as part of the algorithmic verification of critical counts and, implicitly, for decomposing rigid and flexible regions in the framework graph is also represented by a directed graph (see Chapter 3). Directed graphs also appear in the practice of mechanical engineering when synthesizing and analyzing linkages. We will use the directed graphs and their decompositions described in this section to obtain the d -Assur decomposition of a pinned isostatic framework in d -space in Section 7.4.

We first state some basic definitions and background from the theory of directed graphs and present the decomposition of a directed pinned graph into strongly connected components and develop a few simple extensions to confirm the invariance of the decomposition under some natural variations of the directed graph.

7.3.1 Strongly connected component decomposition

We recall a few definitions from Chapter 2 and we define several new terms. Given a *graph* $G = (V, E)$ with a vertex set V and an edge set E , where E is a collection of unordered pairs of *vertices* called the *edges* of the graph. We define a *direction assignment* \vec{G} to graph $G = (V, E)$ as a pair of maps $\text{init}: E \rightarrow V$ and $\text{ter}: E \rightarrow V$ assigning to every edge e an *initial vertex* $\text{init}(e)$ and a *terminal vertex* $\text{ter}(e)$. The edge e is said to be *directed out* of $\text{init}(e)$ and *into* $\text{ter}(e)$. We refer to \vec{G} as a *directed graph* associated with G . When a directed graph has multiple directed edges between the same two vertices, v and w , such graphs are directed *multi-graphs*.

Remark. For simplicity most definitions and proofs are presented in terms of the vocabulary of simple graphs. However, every proof in this chapter will also apply to

multi-graphs. The functions $\text{init}:E \rightarrow V$ and $\text{ter}:E \rightarrow V$ work well in this multi-graph setting, where an edge is no longer uniquely represented by an ordered pair. ■

A *cycle* of a graph G is a subset of edges of G which forms a path such that the start vertex and end vertex are the same. A *directed cycle* is an oriented cycle such that all directed edges are oriented in the same direction along the cycle. A directed graph \vec{G} is *acyclic* if it does not contain any directed cycle. Recall, that the *out-degree* of a given vertex in a directed graph is the number of edges directed out of that vertex. A vertex which has out-degree 0 is called a *sink*. Sink vertices will be *pinned vertices* and directed graphs that have some pinned vertices will be called *pinned graphs*. Pinned graphs are central in this chapter, as they arise in the scheme of a linkage (Figure 7.3(b)) and the associated pinned isostatic graphs in (c) - (e), which will be decomposed later in the chapter.

A directed graph is called *strongly connected* if and only if for any two vertices i and j in \vec{G} , there is a directed path from i to j and from j to i . The *strongly connected components* of a graph are its maximal strongly connected subgraphs. It is maximal in the sense that the subgraph cannot be enlarged to another strongly connected subgraph by including additional vertices and its associated edges (see Fig 7.4 (b), (c)). If a directed graph \vec{G} has n strongly connected components, we can label them by integers $1, 2, \dots, n$, such that a directed edge of \vec{G} always goes from a component of greater number to a smaller number [45]. This labelling is a linear order extending the partial order ⁶. One can determine the strongly connected components of a directed graph using the $O(|E|)$ Tarjan's algorithm [182], for instance, which is implemented in several computer algebra packages, such as Maple, Mathematica and SAGE.

⁶Each directed acyclic graph gives rise to a partial order \leq on its vertices. A partial order is binary relation \leq over a set P , which is reflexive ($a \leq a$), antisymmetric (if $a \leq b$, and $b \leq a$ then $a = b$), and transitive (if $a \leq b$ and $b \leq c$ then $a \leq c$, for all a, b and c in P). If P is linearly ordered under \leq (i.e. linear extension of a partial order), then we also have $a \leq b$ or $b \leq a$ (totality) [45].

To illustrate the decomposition process of a pinned directed graph, we first condense (ground) all pinned vertices into a single ground vertex (sink) (Figure 7.4(a, b)). (More detailed definitions of pinned graphs and grounding will be presented in the next section.) After we have identified the strongly connected components, as a simplification, we will ignore the orientation of edges within the strongly connected components, and just keep the orientation of edges between the components (Figure 7.4 (c)). In the final step, we apply condensation to each strongly connected component by contracting it to a single vertex, obtaining an acyclic graph and a partial order (Figure 7.4(d)). We have schematized the partial order, so that there are no multiple edges appearing between any components.

We now offer a simple observation relating directed cycles and strongly connected components (see Figure 7.24), which will be useful in the subsequent sections and later in the chapter when we introduce a pinned pebble game algorithm.

Lemma 7.3.1 *Cycle reversals leave the strongly connected components unchanged, as well as the overall strongly connected decomposition unchanged.*

Proof We assume that we have condensed all the strongly connected components into individual vertices forming a directed acyclic graph. As cycles only occur within the components, reversing any cycle leaves all the incoming and outgoing edges for the components unchanged, so cycle reversal cannot amalgamate two components.

Conversely, we show that reversing a cycle within a component cannot create two components, using a proof by contradiction. Assume that reversing a cycle has created two components. Then reversing the cycle the second time would amalgamate those two components back into one, so we have a contradiction of the previous argument.

We conclude that the entire acyclic graph does not change, and therefore the strongly connected decomposition is unchanged. ■

So far, we were not concerned how we obtained the directed graphs. In the next subsequent sections we will find directed graphs for our linkages, and use it for our decomposition. Ultimately, we will rely on the pebble game algorithm (section 7.6) to generate the desired directed graph for linkages.

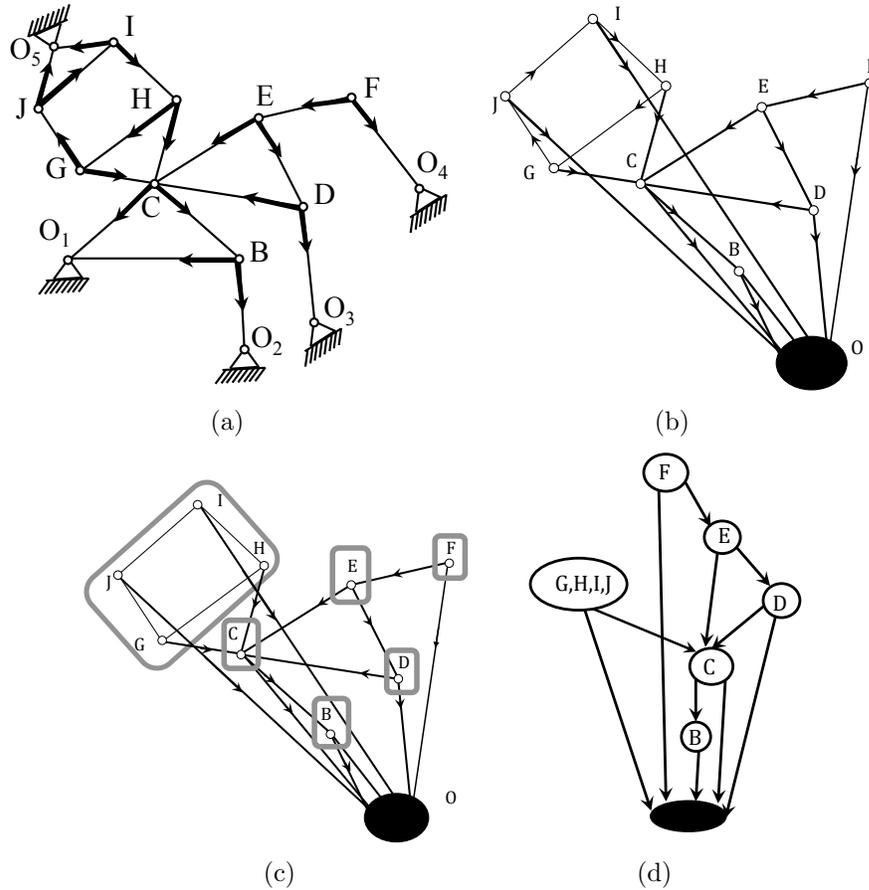


Figure 7.4: Decomposition of a directed pinned graph: The directed pinned graph (a) has the pinned vertices condensed to the ground (sink) (b), with the corresponding strongly connected decomposition (c) and the partial order (d).

7.3.2 Equivalent orientations of a graph

We now show that choices in orientations of edges which conserve a fixed out-degree of each of the vertices do not alter the decompositions. This is a useful observation for the proofs in the next section and will assist the verification of algorithms (i.e.

pebble game) for generating the directed graphs and related decompositions [174]. For example, all plays of the pebble game algorithm with the same number of final pebbles at the corresponding vertices will give the same strongly connected decompositions. We thank Jack Snoeyink for conversations which clarified these arguments.

Definition 7.3.2 *Given a multi-graph G and two direction assignments \vec{G}^1 and \vec{G}^2 . We say that \vec{G}^1 and \vec{G}^2 are equivalent orientations on G if the corresponding vertices have the same out-degree.*

Such alternate orientations appear when comparing direction assignments created by hand (common engineering practice) versus those attained by algorithms (i.e. pebble game, see Section 7.6). We confirm we obtain the same decomposition regardless how we obtained the equivalent direction assignment.

Lemma 7.3.3 *Given two equivalent orientations \vec{G}^1 and \vec{G}^2 on G , then the two orientations differ by reversals on a set of directed cycles.*

Proof Pick an edge $e = (u, v)$ in \vec{G}^1 that is oppositely directed in \vec{G}^2 . So, in \vec{G}^1 edge e is incoming at vertex v . Assume there are k outgoing edges at v in \vec{G}^1 . Because v has same out-degree k in \vec{G}^2 , and this edge is reversed to be outgoing in \vec{G}^2 there must exist an out-going edge from v in \vec{G}^1 , say $f = (v, w)$, that is oppositely directed in \vec{G}^2 (Figure 7.5(a,b)). We walk out of v along f in \vec{G}^1 (Figure 7.5(c)).

As we enter a new vertex, we again apply this same argument. This identifies a directed path in \vec{G}^1 that is oppositely directed in \vec{G}^2 . As there is only a finite collection of edges that have opposite direction, we walk along this directed path until we come back to some vertex on this path, identifying a directed cycle in \vec{G}^1 that has an opposite orientation in \vec{G}^2 .

We reverse the orientation of such a cycle from the orientation in \vec{G}^1 towards the orientation in \vec{G}^2 . This decreases the number of edges in \vec{G}^1 that are oppositely

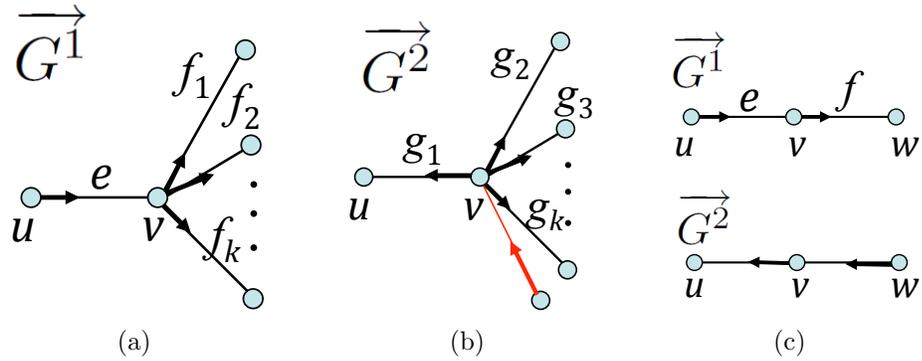


Figure 7.5: Since vertex v has same out-degree in G^1 and in G^2 , there exists an edge in G^1 (a) that is oppositely oriented in G^2 (b). We show such a selection f in (c,d).

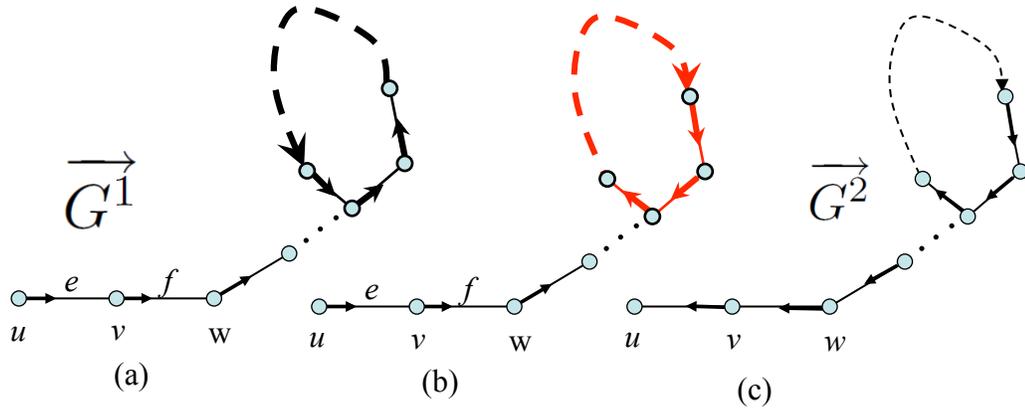


Figure 7.6: Locating a directed path in G^1 that is oppositely oriented in G^2 (a), reversing a first cycle in G^1 (b), which now has the same orientation as in G^2 (c).

directed from $\overrightarrow{G^2}$ (see Figure 7.6). We continue reversing identified cycles, until all edges are directed following $\overrightarrow{G^2}$. ■

We end with a valuable corollary to this lemma (see also Figure 7.24):

Corollary 7.3.4 *Given two equivalent orientations $\overrightarrow{G^1}$ and $\overrightarrow{G^2}$, the strongly connected decompositions are the same.*

Proof From Lemma 7.3.3, $\overrightarrow{G^1}$ and $\overrightarrow{G^2}$ differ by reversal of set of directed cycles. As cycle reversals do not change the strongly connected components the decompositions are the same. ■

7.4 Decomposition of Pinned d -isostatic Graphs

In paper [163], the authors have described the 2-Assur decomposition of a pinned generically isostatic (minimally rigid) graph in the plane. Here we will nuance this decomposition with the connections to the directed graph decomposition of the previous section, and directly extend the central decomposition to all dimensions. This extended decomposition carries significant properties of the 2-Assur decomposition, with one key exception which we examine in section 7.5. Because of applications in mechanical engineering, we are primarily interested in the Assur decompositions in 2-space and 3-space (i.e. 2-Assur and 3-Assur graphs). For this reason all the figures will be either 2 or 3-dimensional.

In this section we extend the definition of 2-Assur graphs to d -space, collectively defining d -Assur graphs. All the results in this section apply to d -Assur graphs. Accordingly, when we just speak of Assur graphs in this section, it should be understood that we are referring to d -Assur graphs. For mathematical completeness, whenever possible we state all the theorems and definitions in d -dimensional space. In section 7.5 we will explore one further property of 2-Assur graphs that does not always extend to higher dimensions - the response of inner vertices to removing one edge.

7.4.1 Pinned d -isostatic graphs and pinned rigidity matrix

Many of the definitions and standard results of rigidity of pinned frameworks that are given here build on from Chapter 2 on regular frameworks, which we will sometimes call unpinned frameworks (or unpinned graphs below). For that reason, we will not provide a detailed discussion. Nevertheless, it is important to be aware of the (slight) differences between pinned and unpinned frameworks, so we will still restate some definitions in terms of the pinned frameworks.

Given a framework associated with a linkage, we are interested in its internal motions, not the trivial ones. It is always possible to transform an unpinned framework to a pinned framework and vice versa (see Figure 7.7). In the plane we can pin any edge of the framework and get a pinned framework (pinning only one vertex does not remove all 3 trivial DOF), or equivalently, by fixing the position of the vertices of some rigid subgraph (i.e. pinning all its vertices). To transform a pinned framework into an unpinned framework we create a minimally rigid (isostatic) subgraph on the pinned vertices. In the plane the easiest construction is to add one edge between any two pinned vertices and add an edge from all the other pinned vertices to both ends of the initial added edge. In 3-space we would construct a triangle between any three pinned vertices and connect all the other pinned vertices with an edge to all three ends of the triangle.

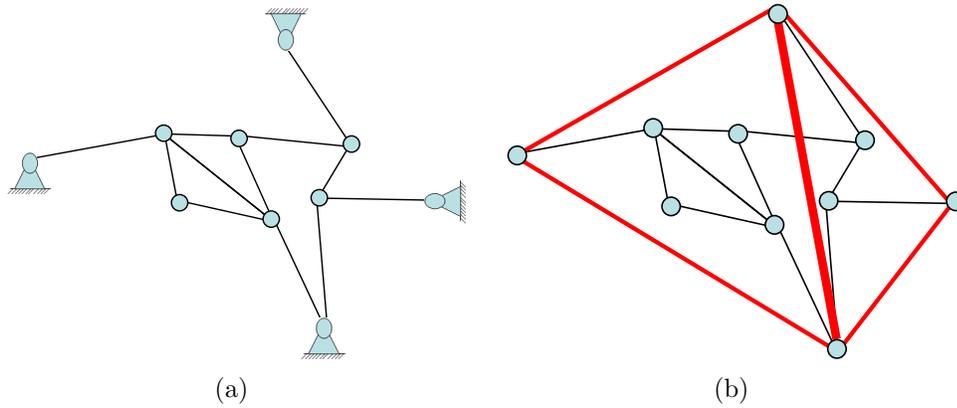


Figure 7.7: The framework (a) is pinned 2-isostatic (minimally rigid) because framework (b) is 2-isostatic. In (b) thick red edge is added between (any) two pins from (a) and an edge is added from all other pins to both ends of this edge. Note that the subgraph on red edges is an isostatic subgraph.

All the discussion in the rest of this chapter is on pinned frameworks and their associated graphs and directed graphs.

More formally, we denote a *pinned framework* as $(\tilde{G}, p) = ((I, P; E), p)$, where $\tilde{G} = (I, P; E)$ is a pinned graph, I is the set of inner vertices (also called unpinned

or free in the linkage literature), P is the set of pinned vertices (i.e. vertices with fixed positions), E is the set of edges, where each edge has at least one endpoint in I , together with an assignment p of points in d -space to the vertices of \tilde{G} (i.e. p is a fixed configuration (embedding) of V into \mathbb{R}^d). Edges that have one inner vertex and one pinned vertex will be called *ground edges*, others will be called *non-ground* or *inner edges*. As pinned vertices never move, edges between pinned vertices are not important to the analysis of a pinned framework and are not part of set E .

We now present a *pinned rigidity matrix*, and give some basic rigidity definitions in terms of a pinned framework (which is very similar to unpinned frameworks as in Chapter 2).

For a pinned framework $(\tilde{G}, p) = ((I, P; E), p)$ we define the $|E| \times d|I|$ d -space *pinned rigidity matrix*, which unlike the regular rigidity matrix for unpinned frameworks (Chapter 2), only has columns for the inner vertices (corresponding to the possible variations in their positions, if the linkage is flexible with the set of (fixed) pinned vertices):

$$\mathbf{R}(\tilde{G}, p) = \begin{matrix} & & & i & & & j & & & \\ & & & & & & & & & \\ & & & & & & & & & \\ \{i, j\} & \left(\begin{array}{cccccccc} \ddots & & & & & & & & \ddots \\ 0 & \dots & 0 & (p_i - p_j) & 0 & \dots & 0 & (p_j - p_i) & 0 & \dots & 0 \\ & & & \vdots & & & & \vdots & & & \vdots \\ \{i, k\} & \left(\begin{array}{cccccccc} 0 & \dots & 0 & (p_i - p_k) & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ & & & \vdots & & & & \vdots & & & \vdots \\ \ddots & & & & & & & \vdots & & & \ddots \end{array} \right) \end{array} \right), \end{matrix}$$

where $i, j \in I$ and $k \in P$. Note that this matrix has d columns for each inner vertex (i.e. each inner vertex has d DOF in dimension d).

The solutions U to the equation: $\mathbf{R}(\tilde{G}, p) \times U^{tr} = 0$ are *infinitesimal motions* of the pinned framework (see Chapter 2 for details). A framework (\tilde{G}, p) is *pinned*

d -rigid if the only infinitesimal motion is the zero motion (i.e. no vertex can move) or equivalently, if pinned rigidity matrix $\mathbf{R}(\tilde{G}, p)$ has maximum rank $d|I|$. A framework is *pinned d -independent* if the rows of $\mathbf{R}(\tilde{G}, p)$ are independent. A framework (\tilde{G}, p) is *pinned d -isostatic* (i.e. pinned minimally rigid) if it is both *pinned d -rigid* and *pinned d -independent*. In particular, if the framework is pinned d -isostatic then $|E| = d|I|$ and the pinned rigidity matrix is a square matrix.

If we vary the configuration p over all of $\mathbb{R}^{d|I|+d|P|}$, then the pinned rigidity matrix achieves some maximal rank, and this maximal rank occurs for an open dense subset of $\mathbb{R}^{d|I|+d|P|}$ - the *generic rank* of the d -space *pinned rigidity matrix* for the graph [203]. In particular, for a pinned isostatic graph, the configurations that drop the rank are captured by a non-zero polynomial in variables for the vertices. If one configuration p achieves the full rank $d|I|$, then almost all configurations (all points in this open dense subset) achieve this rank [203], and we call all configurations in the open dense subset *generic* or *regular*.

We say a pinned framework (\tilde{G}, p) is *generic* if the configuration p of the joints p is generic.

Theorem 7.4.1 *Given a pinned graph $\tilde{G} = (I, P; E)$, the following are equivalent:*

1. *There exists a pinned d -isostatic realization p of \tilde{G} in d -space;*
2. *For all placements $p|_P$ of the pins P in generic position in d -space, and all generic positions of vertices in I the resulting pinned framework is pinned d -isostatic;*
3. *$\tilde{G} = (I, P; E)$ is generically pinned d -rigid and $|E| = d|V|$;*
4. *$\tilde{G} = (I, P; E)$ is generically pinned d -independent and $|E| = d|V|$.*

Proof These statements are translations of standard results for isostatic frameworks to the pinned d -isostatic setting [203]. ■

We call any graph $\tilde{G} = (I, P; E)$ satisfying the equivalent conditions of Theorem 7.4.1 *pinned d -isostatic*.

We recall, that our goal is to turn a mechanical linkage into a pinned d -isostatic graph and decompose it into d -Assur components. In Mechanical engineering literature, Assur graphs (groups) are the said to be “pinned structures of zero mobility that does not contain a simpler structure of same mobility” [138].

We can now give a precise definition of the d -Assur graphs. A *d -Assur graph* is a *minimal* pinned d -isostatic graph. By minimal we mean there is no proper subgraph (with inner vertices) which is also a pinned d -isostatic graph.

To illustrate the definition of Assur graphs in Figures 7.8 and 7.9 we have several examples of 2-Assur and 3-Assur graphs along with graphs that are not Assur. In Figure 7.8 (a - c), these are the most basic Assur graphs in the plane (i.e. 2-Assur), which serve as building blocks of kinematic linkages [162]. For instance, in the literature, the 2-Assur graph in (a) is called a *dyad* and in (b) a *triad*. We have also shown an initial indication of what the decomposition of pinned d -isostatic graphs that are not d -Assur into individual d -Assur graphs will look like. We will shortly look at the decomposition process into d -Assur graphs in more detail.

In section 7.5, we will define a strongly d -Assur graph as a d -Assur graph with the added property that removal of any edge puts all inner vertices in motion. We will look at the difference between the two types of graphs in more detail, but note no distinction between 2-Assur graphs and strongly 2-Assur graphs (see section 7.5), so they will always be called 2-Assur. See also section 7.6 for fast decomposition using the pinned pebble game algorithm.

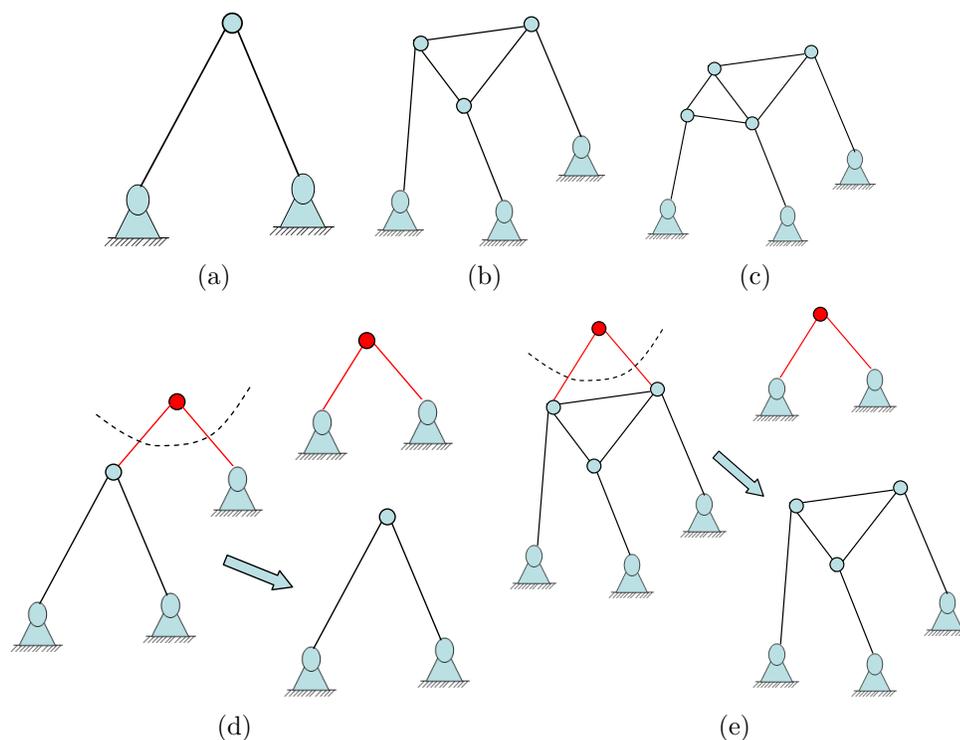


Figure 7.8: Differentiating between pinned isostatic graphs that are 2-Assur and those that are not as they are not 'minimal' pinned isostatic. In (a - c) we have 2-Assur graphs – minimally pinned 2-isostatic (minimally rigid) graphs. In (d) and (e) we have pinned 2-isostatic graphs that are not 2-Assur, as they are not minimally pinned 2-isostatic. The smaller proper subgraph which is 2-Assur is pointed out with an arrow in (d) and (e). Pinned isostatic graphs that are not 2-Assur, can be decomposed into individual 2-Assur graphs as shown in (d) and (e), see text for details of the decomposition process.

7.4.2 Directed graph decomposition of the pinned rigidity matrix

We now use the d -space pinned rigidity matrix to generate a special type of directed graph for a pinned graph in d -dimension with $|E| = d|V|$. We then connect the corresponding directed graph decomposition from section 7.3 to a block decomposition of the pinned rigidity matrix.

Matching the shape of the pinned rigidity matrix, we develop d -directed orientations of the pinned graph: one in which each inner vertex has out-degree d and

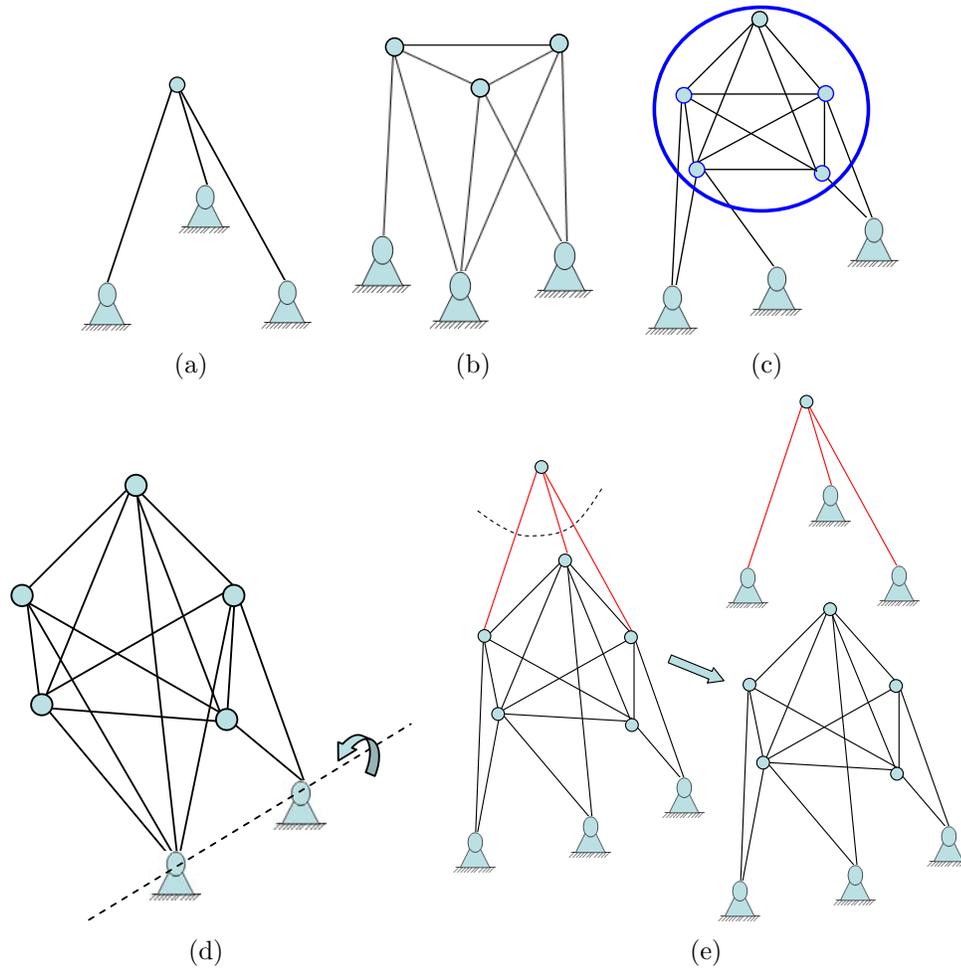


Figure 7.9: In (a) and (b) we have two simple examples of 3-Assur graphs. In (c) the pinned graph is not 3-Assur, as it is not even pinned 3-rigid. It has the right number of edges $3(5) = 15$, however the circled subgraph is overconstrained. The pinned graph in (d) is also not 3-Assur; it is not rigid, as it can rotate around the imaginary line between the two pins. In this graph there are not enough pinned vertices. The graph in (e) is pinned 3-isostatic, but it is not 3-Assur. The smaller proper subgraph which is 3-Assur is pointed out with an arrow. Such pinned 3-isostatic graphs will be decomposable into more 3-Assur components.

each pinned vertex has out-degree 0 - a sink in the directed graph. In section 7.6 we will use the pinned pebble game algorithm as a more algorithmically efficient way to generate the directions on the graph.

Proposition 7.4.2 *Every pinned d -isostatic graph with $|E| = d|V|$ has a d -directed orientation, with all inner vertices of out-degree d and all pinned vertices of out-degree 0.*

Proof Take the determinant of the $d|V| \times d|V|$ pinned rigidity matrix. We know the determinant of the square matrix is non-zero if and only if the framework is infinitesimally rigid. For this determinant to be non-zero, there must be a non-zero term in the Laplace expansion of the determinant of this matrix, in $d \times d$ blocks following the d columns for each inner vertex. Take any such non-zero term. This will associate d rows (edges) with each inner vertex i , and we direct these d edges out from vertex i (Figure 7.10). This gives the desired d -directed orientation. ■

We can now apply the strongly connected graph decomposition from section 7.3 to decompose the pinned d -isostatic graph, with the pinned vertices all identified together in the directed graph decomposition. Each strongly connected component is extended to include the outgoing edges from the component. (In Section 7.4.3, we show that these extended components are minimal pinned d -isostatic graphs.) We make the connections through a block-triangular decomposition of the pinned d -rigidity matrix (Figure 7.10). In this decomposition, a permutation of the vertices and the edges of the pinned graph generates a lower block-triangular matrix.

Theorem 7.4.3 *For a pinned d -isostatic graph with a pinned d -directed orientation (all inner vertices of out-degree d and all pinned vertices of out-degree 0), the strongly connected decomposition (with the pinned vertices identified) coincides with the block-triangular decomposition of the pinned rigidity matrix with a maximal number of diagonal blocks, for some linear order of the blocks extending the partial order of the directed graph decomposition.*

Proof Given a pinned d -isostatic graph with a pinned d -directed orientation, we apply the techniques of section 7.3 to the pinned graph with the pinned vertices

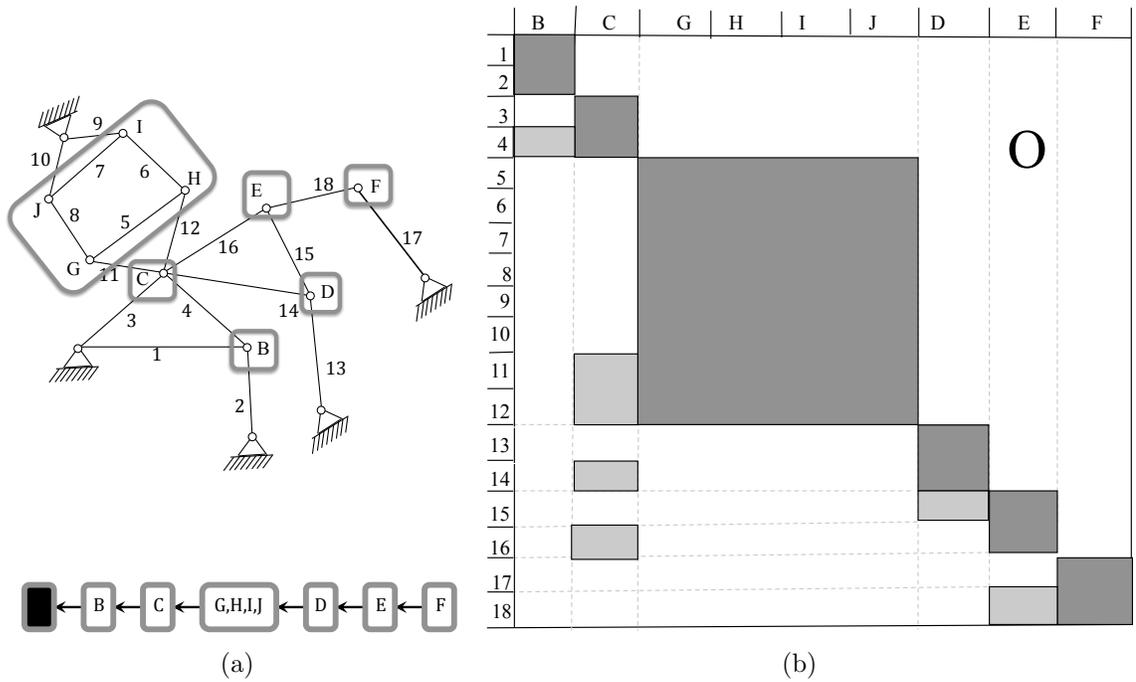


Figure 7.10: The pinned 2-isostatic framework in (a), with the directed graph decomposition of Figure 7.4 as a partial order, extended to a linear order generates a block triangular matrix in (b).

identified to obtain a strongly connected decomposition. If this has more than one component, then focus one of the bottom extended components A along with its edges to the ground. With a permutation of the rows and a permutation of the columns, we place these vertices and edges at the top left of the matrix. (For notation, we assume with $i, j \in A$ and $\ell, m \in B$ and k is a pinned vertex, where B is another component.)

This gives the form:

	...	i	...	j	ℓ	...	m	...
	\ddots		\vdots		\ddots	\ddots		\vdots		\ddots
$\{i, j\}$...	0	$p_i - p_j$	0	...	0	...	0	...	0
$\{i, k\}$...	0	$p_i - p_k$	0	...	0	...	0	...	0
	\ddots		\vdots		\ddots	\ddots		\vdots		\ddots
	\ddots		\vdots		\ddots	\ddots		\vdots		\ddots
$\{i, \ell\}$...	0	$p_i - p_\ell$	0	0	$p_\ell - p_i$	0	...
$\{\ell, m\}$...	0	0	...	0	...	0	$p_\ell - p_m$	0	$p_m - p_\ell$
	\ddots		\vdots		\ddots	\ddots		\vdots		\ddots

or simply:

$$\mathbf{R}(G) = \begin{array}{|c|c|} \hline \mathbf{R}(A) & 0 \\ \hline X & \mathbf{R}(B) \\ \hline \end{array}$$

We have an initial block decomposition. Repeating this for each of the blocks up the acyclic strongly connected decomposition, we find a diagonal matrix block for each component of the decomposition, and the matrix has the desired block-triangular

form:

$$\mathbf{R}(G) = \begin{array}{|c|c|c|c|c|} \hline \mathbf{R}(A_1) & 0 & 0 & \dots & 0 \\ \hline X_{12} & \mathbf{R}(A_2) & 0 & \dots & 0 \\ \hline \vdots & \vdots & \ddots & \ddots & \vdots \\ \hline X_{1n} & X_{2n} & X_{3n} & \dots & \mathbf{R}(A_n) \\ \hline \end{array}$$

Note that the acyclic strongly connected decomposition graph only gives a partial order. The matrix block-triangular decomposition imposes a linear order which extends this partial order. The linear order will not be unique, as it is possible that some blocks may be incomparable in the partial order - which will show up here as some off diagonal lower blocks with $X_{ij} = 0$. (See Figure 7.10 where component $(GHIJ)$ could be moved anywhere to the right in the linear order.)

Conversely, if we have such a block-triangular decomposition, then the corresponding Laplace decomposition of the determinant will produce a pinned d -directed orientation in which all edges are either oriented within the block, or directed towards vertices in blocks to the left. In short, we have a decomposition with an acyclic decomposition graph extracted from the linear order of the blocks .

If there is a further decomposition from the d -directed graph, then this will show up as a further block-triangular decomposition with more components, and vice versa. ■

Recall that any two pinned d -directed orientations are equivalent (Section 7.3.2), so they give the same decomposition in the algorithm of section 7.3 (see Figure 7.4). Therefore in the block-triangular decomposition of the pinned rigidity matrix,

the linear order extends this partial order. Note in particular, each other non-zero term in the Laplace block expansion gives an equivalent pinned d -directed orientation, and an equivalent block-triangular decomposition of the pinned rigidity matrix.

7.4.3 d -Assur graphs and minimal components

We now focus on the minimal components in these decompositions of a pinned d -isostatic graph $\tilde{G} = (I, P; E)$. We will verify that these are minimal pinned d -isostatic components of \tilde{G} , i.e. the d -Assur graphs. We also verify some key properties of these graphs.

Theorem 7.4.4 (d -Assur Graphs) *For a pinned d -isostatic graph \tilde{G} the following are equivalent:*

1. *the graph contains no proper pinned d -isostatic subgraphs (i.e. d -Assur);*
2. *the graph is indecomposable for some (any) d -directed orientation (i.e. \tilde{G} has one strongly connected component on inner vertices);*
3. *the pinned rigidity matrix has no proper block triangular decomposition.*

Proof Theorem 7.4.3 shows the equivalence of (2) and (3). It remains to show that (1) is equivalent to these.

Assume the graph is not minimal and there is a proper pinned isostatic subgraph \hat{G} . We will show the pinned rigidity matrix decomposes. If we permute the inner vertices and all the edges of \hat{G} to the upper left corner of the pinned rigidity matrix, the rest of these rows from \hat{G} are 0 and the remaining columns and rows form a second block. This gives a block triangular decomposition of the pinned rigidity matrix. The contrapositive says that if the pinned rigidity matrix does not have a proper block triangular decomposition, then the pinned isostatic graph is minimal.

Conversely, if the pinned rigidity matrix for \tilde{G} has a proper block triangular decomposition, then it is clear the upper left block represents the inner vertices and the edges (including the ground edges) of a proper pinned isostatic subgraph \hat{G} . The contrapositive confirms that if the graph \hat{G} is minimal then the pinned rigidity matrix does not decompose. ■

Any graph \tilde{G} which satisfies one of these three equivalent properties is *d-Assur*, so we have additional ways to describe a *d-Assur* graph \tilde{G} . We can use the concept of a minimal pinned isostatic graph (*d-Assur* graph) to build up a third route to the decomposition of a larger pinned *d-isostatic* graph into *d-Assur* graphs (*Assur* components). We have already given an initial decomposition of a 2-isostatic (and 3-isostatic) graph into 2-Assur (3-Assur) components in Figures 7.8 and 7.9.

We outline the general process of decomposition, which matches what we already have from Theorem 7.4.3 (refer to Figure 7.11).

We start with the ground as the bottom layer. Find a minimal *d-isostatic* pinned subgraph (*d-Assur* component). This will be above the ground component. (In Figure 7.11 (a) this is the subgraph on vertices (B, C, D, E, F) and all the edges down to the ground). Combine that into the ground, and seek another minimal *d-isostatic* pinned subgraph (*d-Assur* component). This will be a component above all components to which its ground edges attach. Repeat until all vertices are in some component. This is the *d-Assur decomposition* of the pinned *d-isostatic* graph \tilde{G} .

7.4.4 Summary Assur Decomposition

We now pull these connections into a summary theorem about the three equivalent ways to decompose a pinned *d-isostatic* graph.

Theorem 7.4.5 (*d*-Assur Decomposition) *Given a pinned d -isostatic graph \tilde{G} , there is a d -directed orientation of \tilde{G} . With any such d -directed orientation, the following decompositions are equivalent:*

1. *the d -Assur decomposition of \tilde{G} ;*
2. *the strongly connected decomposition into extended components associated with the d -directed orientation (with all pins identified);*
3. *the block-triangular decomposition of the pinned rigidity matrix into a maximal number of components for some linear order extending the partial order of (i) or equivalently (ii).*

Proof Theorem 7.4.3 give the equivalence of (2) and (3). The equivalence of (1) and (2) follows from Theorem 7.4.4. The construction process for the d -Assur decomposition above guarantees that for two components A, B A is above B in the d -Assur partial order if and only if A is above B in the d -directed partial order. ■

7.4.5 Pinned d -counts and insufficiency of d -directions

While having a d -directed orientation is a necessary condition for being pinned d -isostatic, it is not sufficient. These orientations are also connected to some basic necessary counting conditions - conditions that have been proven sufficient in the plane (i.e. Laman's theorem), but fail to be sufficient in 3-space. Since there have been efforts to transform this necessary condition into a sufficient condition, we give a few key observations and examples to clarify the connections.

Theorem 7.4.6 (Necessary Pinned d -Counts) *Given a pinned d -isostatic graph $\tilde{G} = (I, P; E)$, the following properties hold:*

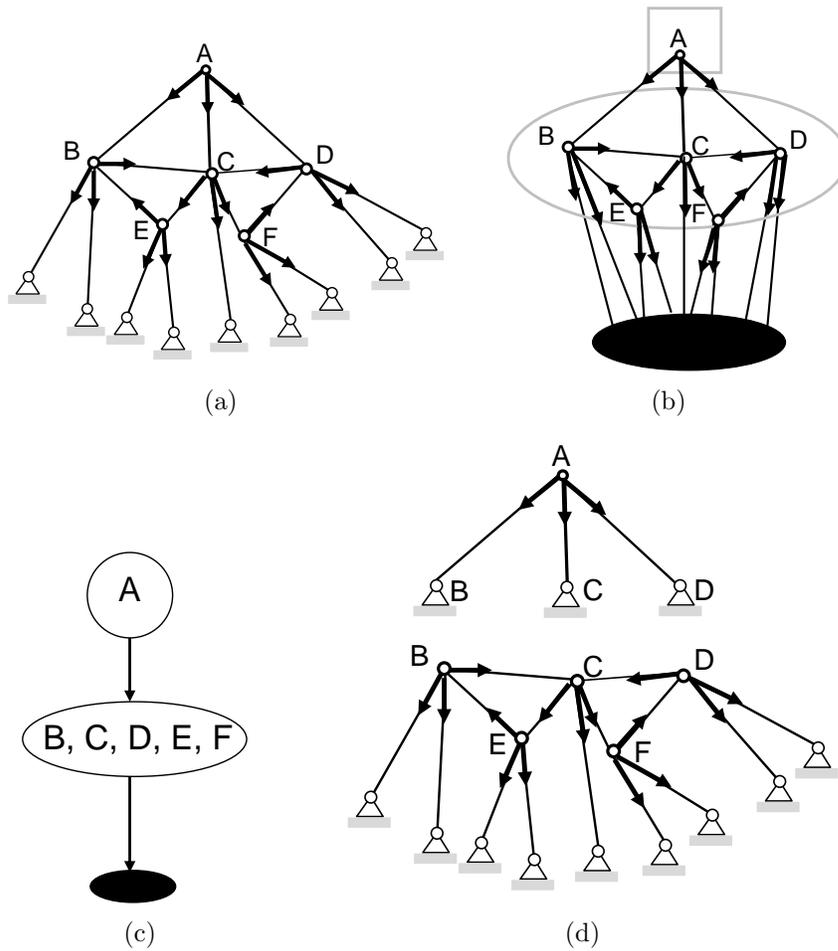


Figure 7.11: A pinned 3-isostatic framework (a) has a unique decomposition into 3-Assur graphs through condensing the pins to ground (rigid region) (b) and creating a partial order (c) which also corresponds to a scheme of (individual) 3-Assur graphs (d).

(1) $|E| = d|I|$.

(2) for all subgraphs $\tilde{G}' = (I', P'; E')$

(i) $|E'| \leq d|I'|$ if $|P'| \geq d$,

(ii) $|E'| \leq d|I'| - \binom{d+1-k}{2}$ if $|P'| = k$, $1 < k < d$,

(iii) $|E'| \leq d|I'| - \binom{d}{2}$ if $|P'| = 1$ and

(iv) $|E'| \leq d|I'| - \binom{d+1}{2} = d|I'| - (d + \binom{d}{2})$ if $P' = \emptyset$.

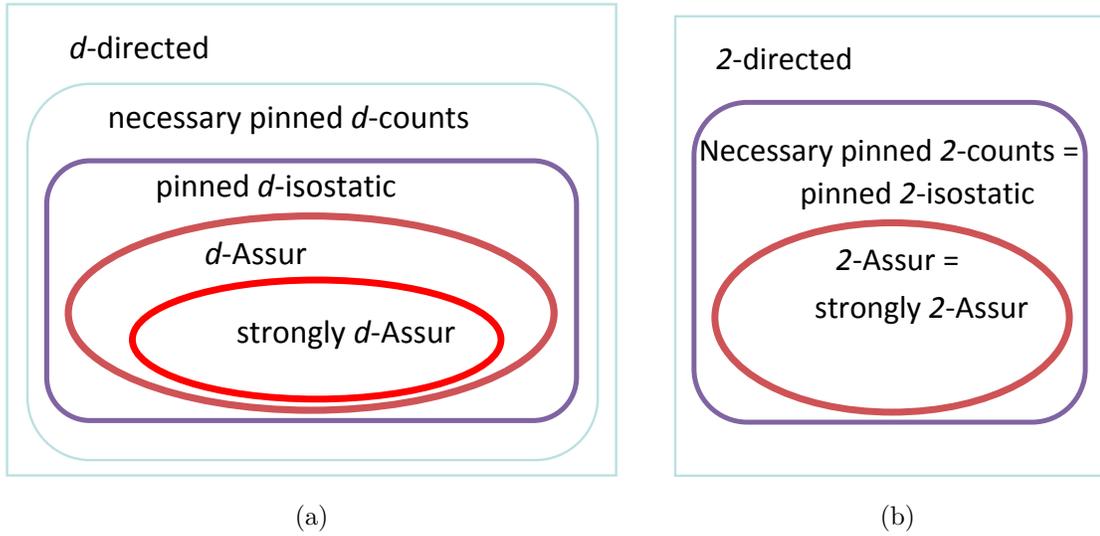


Figure 7.12: (a) shows the general nesting of conditions for dimension d , including strongly d -Assur graphs, while (b) shows the simplified nesting in the plane.

Proof The count (1) is just the condition for the pinned d -rigidity matrix to be square. For the subgraphs, the count 2(i) is again the subgraph count on the submatrix of the columns of I' , whose failure would guarantee a row dependence.

In 2(ii) $\binom{d+1-k}{2}$ is the count of remaining rotational degrees of freedom with k points pinned. For instance, with $d = 3$ and $k = 2$ ($|P'| = 2$), there must be a $\binom{2}{2} = 1$ remaining degree of freedom, corresponding to a rotation of the entire framework about a line passing through these two points. Note that pinning one point eliminates all d translational degrees of freedom. As we continue to pin additional vertices we are eliminating more rotational degrees of freedom. With k points fixed, $k - 1$ dimensional space is fixed. The rotational axis in any dimension d is of dimension $d - 2$ (i.e. point in plane, line in $3D$, plane in $4D$, etc). The possible rotations occur in the orthogonal complement to the rotational axis, which is a 2-dimensional space. With k points fixed from the remaining $d - (k - 1)$ space we choose all such possible 2-dimensional orthogonal complements which gives us the remaining $\binom{d-(k-1)}{2} = \binom{d+1-k}{2}$ space of rotations (DOF). For 2 (iii), there is a space of $\binom{d}{2}$ rotations fixing the single vertex in

dimension d , and for 2(iv) there is a $\binom{d+1}{2}$ -space of trivial motions (combination of d translations and $\binom{d}{2}$ rotations), so the bounds on the count of edges is the maximum number of independent rows. ■

Note that the formula for the subgraph count in 2(ii) of Theorem 7.4.6 is also the correct form for all $k \geq 0$.

In dimension 2, the necessary pinned 2-counts become (*Pinned Plane Framework Conditions* for $\tilde{G} = (I, P; E)$) [163]:

1. $|E| = 2|I|$ and
2. for all subgraphs $\tilde{G}' = (I', P'; E')$ the following conditions hold:
 - (i) $|E'| \leq 2|I'|$ if $|P'| \geq 2$,
 - (ii) $|E'| \leq 2|I'| - 1$ if $|P'| = 1$, and
 - (iii) $|E'| \leq 2|I'| - 3$ if $P' = \emptyset$.

For completeness, the Necessary pinned 3-counts are:

1. $|E| = 3|I|$.
2. for all subgraphs $\tilde{G}' = (I', P'; E')$
 - (i) $|E'| \leq 3|I'|$ if $|P'| \geq 3$,
 - (ii) $|E'| \leq 3|I'| - 1$ if $|P'| = 2$, and
 - (iii) $|E'| \leq 3|I'| - 3$ if $|P'| = 1$.
 - (iv) $|E'| \leq 3|I'| - 6$ if $P' = \emptyset$.

A useful observation is that all pinned graphs in dimension d that satisfy the necessary pinned d -counts can be d -directed:

Theorem 7.4.7 *If a pinned graph $\tilde{G} = (I, P; E)$ satisfies the Necessary pinned d -counts for some d , then it has a d -directed orientation with all inner vertices of out-degree d and inner vertices of out-degree 0.*

Proof Since the graph satisfies the Necessary pinned d -counts, we can play the $d|V|$ pebble game on both inner edges and ground edges (all subgraphs satisfy the necessary pinned d -counts, so no special pebble acceptance criteria is needed). Start by placing d free pebbles on each inner vertex. Cover edges (inner edges and ground edge) with a pebble in any order. All $d|V|$ pebbles will be used up (there are $d|V|$ edges) so each inner vertex will be d -directed, giving us the desired d -directed orientation of the graph. (See section 7.6 for a pinned version of the pebble game in the plane, also [174].) ⁷ ■

Having stated the necessary pinned d -counts, we can now illustrate that d -directed orientation is not a sufficient condition for being pinned d -isostatic. This is not surprising given that we do not have both necessary and sufficient counting conditions in dimensions higher than 2 (see below for more commentary). In Figure 7.13 (a) we have an example of a graph which is 2-directed but not 2-isostatic. It is not 2-isostatic as it fails the necessary pinned 2-counts. The example in Figure 7.13 (b) shows a similar example in 3-space, it has a proper 3-directed orientation, although it is not pinned 3-isostatic. Note that both graphs have one strongly connected component on the inner vertices. These two examples confirm that having a 2 and 3-directed orientation and being indecomposable does not capture the very basic subgraph counting conditions of Theorem 7.4.6.

Remark. Another way to tell that the pinned graphs in Figure 7.13 are not even rigid is that in 2-space (3-space) we need at least three (six) ground edges to attach

⁷Note that this proof could be used as an alternate proof of Theorem 7.4.2 as the assumption of that Theorem was that the graph is pinned d -isostatic, so it automatically satisfies the Necessary pinned d -counts.

the rigid bodies (with at least 2 (3) vertices in 2-space (3-space)) to the ground so the graphs become (generically) pinned 2-rigid (3-rigid), respectively. In (a) the rigid subgraph is attached to the ground with only two ground edges (has only two pins), and in (b) the rigid subgraph is attached to the ground with only 5 ground edges, hence they are not pinned rigid. ■

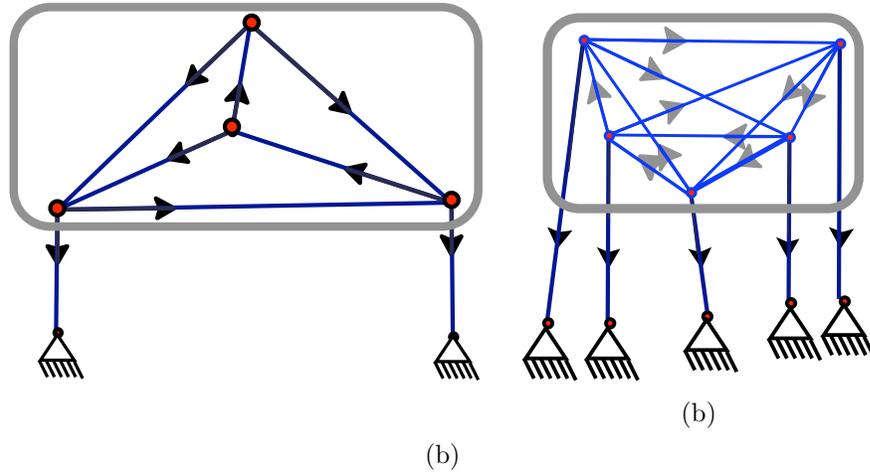


Figure 7.13: A pinned graph which is 2-directed, indecomposable (there is one strongly connected component on the inner vertices) but not 2-isostatic as the subgraph in the grey box does not satisfy the necessary pinned 2-counts. The gray subgraphs is overcounted (i.e. redundantly rigid subgraph - one edges is wasted (redundant)). (a). A pinned graph which is 3-directed, indecomposable but not 3-isostatic (the subgraph in grey box does not satisfy the necessary 3-counts - it is overcounted) (b).

In Figure 7.14 we have another example which has a 3-directed orientation (a), although it is not pinned 3-isostatic (minimally rigid) (see remark below for further discussion on this special example). This graph is decomposable (b), but what is interesting is that one of the components (c) now visibly fails the necessary pinned 3-count though is still 3-directed.

Remark. By translating the general results for generically isostatic (minimally rigid) graphs in dimension 1 and 2, the pinned counts in dimension 1 and 2 are also sufficient for a graph to be pinned 1 and 2-isostatic, respectively. In dimension 2, the necessity and sufficiency of the counts is captured in the Pinned Laman Theorem [163]:

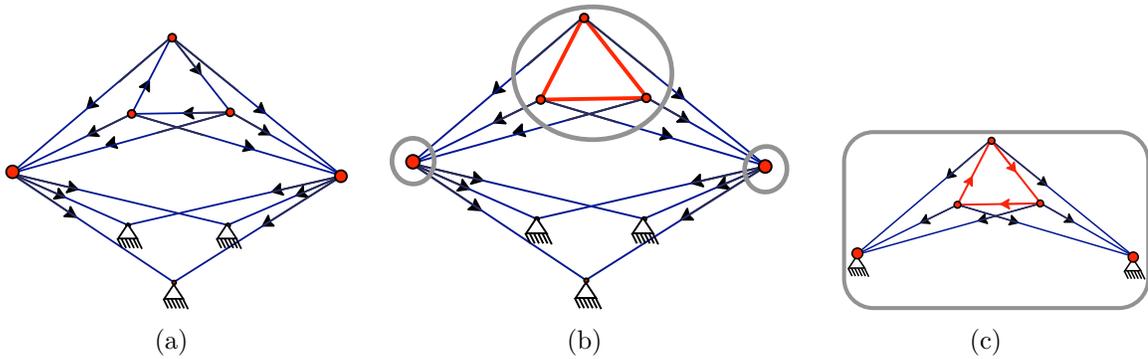


Figure 7.14: A pinned graph which is 3-directed (a), has a strongly connected decomposition (b) but the top extended component (c) is not 3-isostatic.

Theorem 7.4.8 (Pinned Laman Theorem[163]) *Given a 2-dimensional pinned graph $\tilde{G} = (I, P; E)$. \tilde{G} is (generically) pinned 2-isostatic (minimally pinned rigid) if and only if \tilde{G} satisfies the Pinned Plane Framework Conditions (i.e. Necessary Pinned 2-counts - as given above).*

We have seen in chapter 2 that there is no necessary and sufficient counting condition in dimension $d > 2$ in unpinned bar and joint frameworks (graphs). Similarly, the necessary pinned counting conditions are not sufficient for pinned d -isostatic graphs. The example in Figure 7.15 (a) (analogue to the ‘double banana’ example in 3D unpinned bar and joint frameworks - there is a double banana graph in the strongly connected component encircled in blue) shows that the necessary pinned 3-counts are not sufficient for rigidity in dimension 3. In Figure 7.15 (b) we also have an example of a graph which shows that counts are not sufficient in dimension 4. ■

A d -directed graph has an immediate directed graph (strongly connected) decomposition. Whether the graph is pinned d -isostatic will depend on whether all of the individual extended components are pinned d -isostatic. So, it could still be useful to decompose this pinned graph (i.e. strongly connected components with the ends of the extended edges as pinned vertices, as it would be easier to analyze the rigidity/flexibility of individual small components than the entire pinned graph.

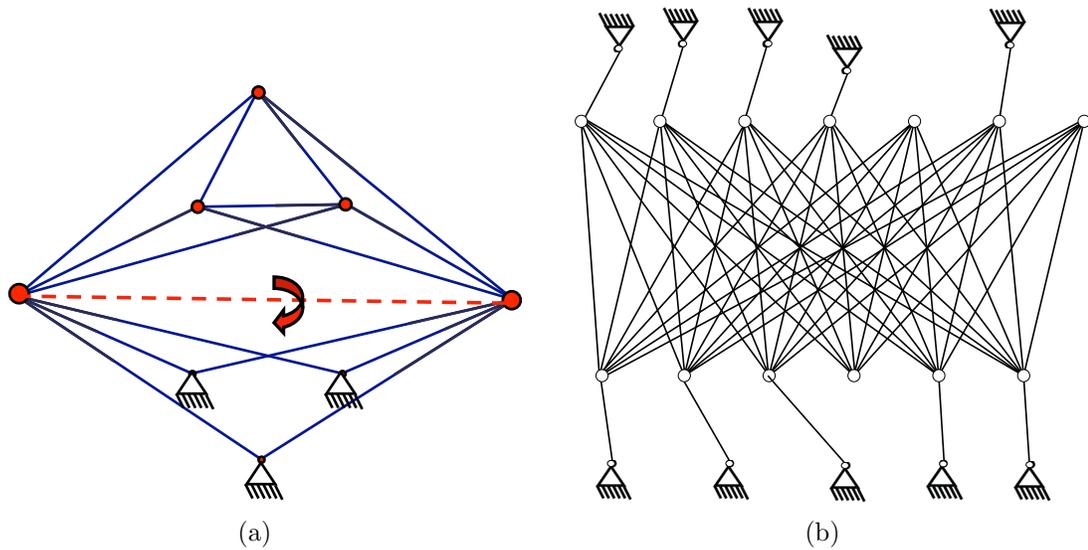


Figure 7.15: A pinned graph in $3D$ (a) which satisfies the necessary pinned 3-counting conditions of Theorem 7.4.6 but is not rigid . Pinned $K_{6,7}$ graph (b) satisfies the pinned 4-counts, but is not pinned 4-isostatic [205].

While a directed graph decomposition detects some failures, via failed necessary counts on subgraphs (see Figure 7.14), detecting from a graph if a d -pinned framework is d -isostatic for $d > 2$ is generally difficult. Alternatively, we would have to resort to analysis of the pinned rigidity matrices. A further difficulty in $d > 2$ is illustrated in Figure 7.16 which shows a 3-directed indecomposable graph that satisfies all the subgraph necessary pinned 3-counts of Theorem 7.4.6 but is still not pinned 3-isostatic. Thus, even combined with the d -Assur decompositions, we do not have a necessary and sufficient counting conditions for a graph to be d -Assur when $d > 2$. One could easily detect the failure of the type in Figure 7.13 using a pinned pebble game procedure (see section 7.6) (those graphs do not satisfy the necessary pinned counts of Theorem 7.4.6 on all subgraphs), but there are no known polynomial algorithms for the failures of type Figure 7.16 [114, 174, 203].

Since in 2D we have both necessary and sufficient counting conditions for pinned 2-isostatic graphs (Pinned Laman’s Theorem 7.4.8) it is easier to check whether any pinned graph is 2-Assur (Figure 7.13(b)). We need to test both (i) the complete

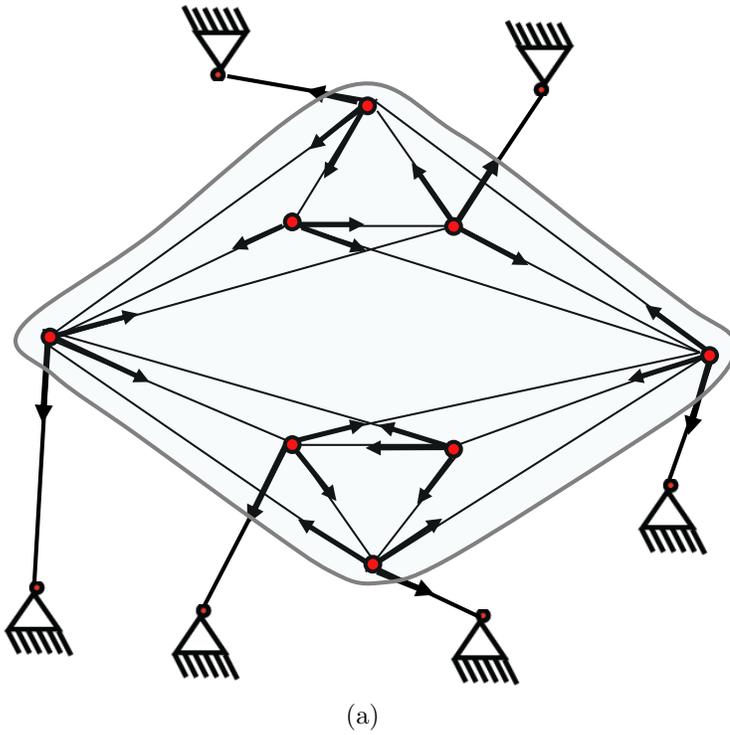


Figure 7.16: Pinned graph which is 3-directed, indecomposable, satisfies the necessary pinned 3-count, but it is not 3-isostatic.

set of Pinned Plane Framework Conditions (Pinned Laman's Theorem) and (ii) the indecomposability. Given the 2-directed orientation of a pinned graph in the plane, the decomposition algorithm for strongly connected components is linear in $|E|$ [182]. The use of the rigidity matrix to generate the directions is not in practice very useful, and was used in section 7.4.2 mainly in proofs. In section 7.6 we will show how to use the pinned pebble game algorithm to efficiently extract the direction on the graphs, and when combined with the strongly connected algorithm (such as Tarjan's), gives us a fast algorithm to test both (i) and (ii), and obtain the 2-Assur decomposition.

7.5 Tracing Motions in Linkages through Assur Decompositions

In the context of linkages, a central property is how an associated framework responds when one of the links (edges) is replaced by a ‘driver’ so that this distance between the end points is controlled by a piston, or we control an angle at a hinge, as in a robot arm. In terms of rigidity we can translate this, and ask: What vertices go in motion if one of the edges (or vertices) is removed from a pinned d -isostatic graph? We will word the theorems whenever possible for d -Assur graphs, though several key results only hold for 2-Assur graphs. Because of applications we again restrict ourselves to the examples only in dimensions 2 and 3. In this section we will also distinguish the special class of *strongly d -Assur graphs* by the feature that all inner vertices go into motion for every choice of inserted driver (i.e. removal of any edge). This is the direct extension of a stronger key property of 2-Assur graphs [163] for which there is no distinction from the strongly 2-Assur graphs. This is not a matter of a deeper decomposition - but a difference in the characteristics of the underlying components (i.e. are components strongly d -Assur or d -Assur) we are analyzing (or synthesizing).

7.5.1 Motions generated by removing an edge:

d-Assur vs strongly *d*-Assur graphs

The following result generates a 1 DOF linkage by removing an edge from an d -isostatic framework. Recall that an edge is part of a unique d -Assur graph in the extended decomposition, in which each strongly connected component is extended to include its outward directed edges. Recall that we defined a *strongly d -Assur graph* a d -Assur graph (minimal pinned d -isostatic graph) with the added property

that removal of any edge induces a motion of all the inner vertices. In the plane all 2-Assur graphs are strongly 2-Assur graphs, as we will show.

Proposition 7.5.1 *If \tilde{G} is a pinned d -isostatic graph, and at a generic configuration p , removal of any edge from \tilde{G} causes an infinitesimal motion which is non-zero at all its inner vertices, then \tilde{G} is strongly d -Assur.*

Proof We need to show that \tilde{G} is a ‘minimal’ pinned d -isostatic graph (i.e. there is no proper subgraph which is also a pinned d -isostatic graph - recall Figures 7.8 and 7.9). By assumption \tilde{G} is a pinned d -isostatic graph. Assume \tilde{G} is not minimal pinned d -isostatic. Then by Theorem 7.4.4 there is a block-triangular matrix decomposition with more than one block. Removing an edge from any block which is lower right in the matrix will leave the graph associated with the upper left block (equivalently at the bottom of the directed graph decomposition) as pinned d -isostatic. This guarantees that the solution $R(\tilde{G}, p) \times U = 0$ is zero on all vertices of this upper block (i.e. these vertices have no motion), a contradiction. Therefore \tilde{G} must be a minimal pinned d -isostatic graph. Since \tilde{G} is minimal and by assumption removal of any edge from \tilde{G} causes all inner vertices to go into motion, \tilde{G} is strongly d -Assur. ■

Note that in Proposition 7.5.1, we could have assumed that \tilde{G} is pinned d -rigid, and the same conclusion would still follow. The assumption of independence was not necessary, as removal of any edge which causes a motion in the graph, indicates that this edge is independent.

We have the following stronger statement about 2-Assur graphs (see Figure 7.12(b) for relationship between graphs in dimension 2):

Proposition 7.5.2 *If we remove any edge from a 2-Assur graph \tilde{G} then this leaves a graph which, at a generic configuration p has an infinitesimal motion which is non-zero at all inner vertices. Therefore \tilde{G} is strongly 2-Assur.*

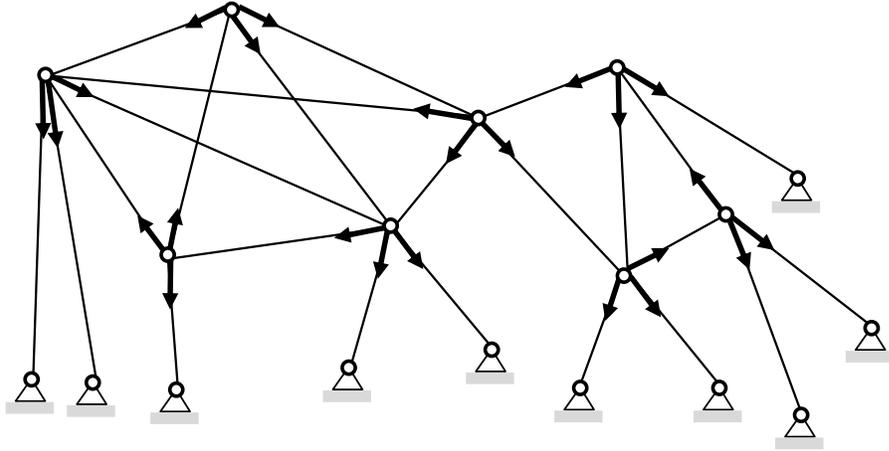


Figure 7.17: An example of a strongly 3-Assur graph. Removal of any ground edge or inner edge induces a motion of all inner vertices.

Proof Assume the graph \tilde{G} is 2-Assur. This means that $|E| = 2|I|$. Removing any one edge will leave the pinned rigidity matrix with $|E| = 2|I| - 1$, so there must be a nontrivial solution U to the matrix equation $R(\tilde{G}, p) \times U = 0$. If $U_i = 0$ on some inner vertex i , then that vertex is still rigidly connected to the ground, and therefore must be in a pinned subgraph \tilde{G}' with $|E'| = 2|I'|$. This subgraph \tilde{G}' would itself be a pinned 2-isostatic graph which could not include at least one of the vertices of the removed edge. Such a subgraph contradicts the minimality of the original 2-Assur graph \tilde{G} . ■

This proposition confirms that 2-Assur graphs are equivalent with strongly 2-Assur graphs, which we restate as a corollary:

Corollary 7.5.3 *Every 2-Assur graph \tilde{G} is a strongly 2-Assur graph.*

The analog of Proposition 7.5.2 in d space ($d > 2$) fails, and there are explicit counter-examples. The example in Figure 7.17 is a strongly 3-Assur graph since removal of any edge causes all inner vertices to be mobile, while the examples in Figure 7.18 are only 3-Assur. We should point out that all the examples that were presented in section 7.4 were of strongly 2 or 3-Assur graphs.

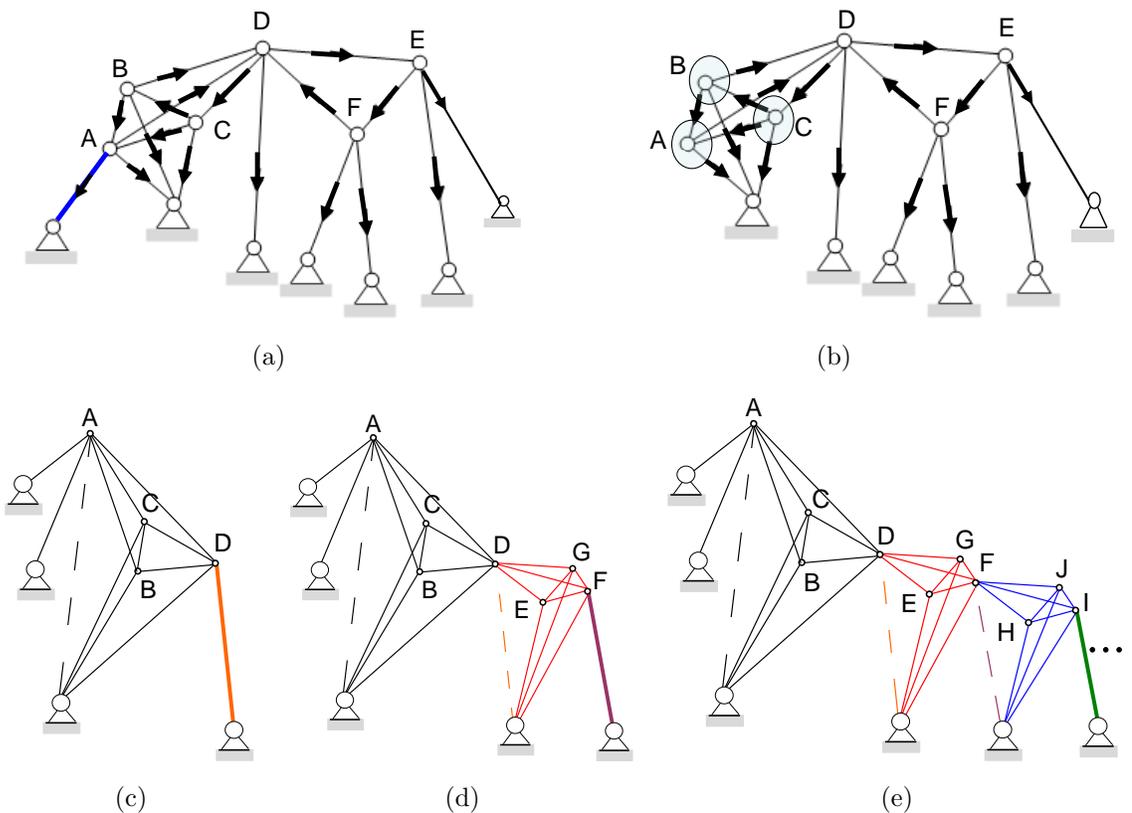


Figure 7.18: Examples of 3-Assur graphs which are not strongly 3-Assur. Removal of a blue edge (a) results in a motion of only the circled vertices in (b). In the examples in (c, d, e), we introduce a new copy of a banana graph, and illustrate how removal of certain edges induces smaller sets of vertices to move. Removal of an orange edge in (c) causes all vertices to move except A, which is held rigidity to the ground by an implicit (imaginary) edge indicated with a dashed line. This process is continued through (d,e). In (d) removal of the purple edge at F will only cause E, F and G to move, and removal of any red edge will cause all vertices but A to move. Removal of a green edge at I in (e) induces a motion of only vertices I, J and H. Removal of any black edge puts all vertices into motion.

The examples in Figure 7.18 (c, d, e) are particularly interesting. Starting with a 3-Assur graph in (c) we keep constructing new 3-Assur graphs as shown in (d) and (e). In these set of examples, removal of some edges (drivers) induces a full motion of inner vertices while removal of other edges induces motions in a smaller collection of inner vertices. So, given a 3-Assur graph, one might be interested in distinguishing those edges (drivers) whose removal puts all inner vertices into motion

from other edges whose removal puts only some vertices in motion (which we can call *weak drivers*). Since 2-Assur graphs are the same as strongly 2-Assur graphs, we will never observe this in 2-Assur graphs. Any edge in a 2-Assur graph is equivalent in the sense that its removal puts all inner vertices in the motion.

In 2D this distinction among the drivers would only occur in the graphs that are not 2-Assur, where removal of any edge in one component will only cause the motion in that 2-Assur component and all the other 2-Assur components below it in the acyclic 2-Assur decomposition. We will look at this in more detail in section 7.6, when we illustrate the pinned pebble game algorithm.

The fact that not all 3-Assur graphs are strongly 3-Assur is connected to obstacles to a good combinatorial (counting) characterization for 3-dimensional bar and joint frameworks, also noted in the lack of necessary and sufficient counting conditions in higher dimensional frameworks (i.e. lack of a Laman type of a theorem). In this context, detecting the difference between d -Assur graphs and strongly d -Assur graphs can be equally challenging (when $d > 2$). We pose this difficult open problem in combinatorially rigidity:

Given a d -Assur graph ($d > 2$) \tilde{G} , find a combinatorial method (using only the information from the graph) that determines if \tilde{G} is strongly d -Assur.

Remark. Despite these difficulties in higher dimensional space, having a comprehensive understanding of 2-dimensional Assur graphs can be very useful to the analysis of 3-dimensional linkages. A frequent practice by mechanical engineers and in robotics community is to decompose the 3-dimensional structure to several 2-dimensional components, which significantly simplifies the analysis.

In many cases, 3-dimensional linkages are built up by carefully connected copies of planar structures. Often identical 2-dimensional structures are reused forming

the larger 3-dimensional structure, where the motions of individual 2-dimensional structures are restricted to the plane by the geometry of the other constraints. ■

Recall from chapter 2 (i.e. Gluck's Theorem), a standard result in rigidity theory says [65, 203] for generic configuration p , infinitesimal rigidity/flexibility (infinitesimal motions) coincides with rigidity/flexibility (finite motions). Thus, we can summarize the two propositions from this section in terms of finite motions in the following corollaries:

Corollary 7.5.4 *If \tilde{G} is a pinned d -isostatic graph, and at a generic configuration p , removal of any edge from \tilde{G} causes a finite motion which is non-zero at all its inner vertices, then \tilde{G} is strongly d -Assur.*

Corollary 7.5.5 *If we remove any edge from a 2-Assur graph \tilde{G} , then this leaves a graph which, at a generic configuration p has a finite motion which is non-zero at all inner vertices.*

7.5.2 Vertex removal

Another engineering technique used in testing for decompositions is to remove an inner vertex and check which vertices go in motion (i.e. become mobile). Removing an inner vertex also removes edge(s) (any edge will always contain at least one inner vertex), leading to a vertex analog of Theorem 7.5.4.

Theorem 7.5.6 *If \tilde{G} is a pinned d -isostatic graph, and at a generic configuration p , removal of any inner vertex from \tilde{G} causes a finite motion which is non zero at all inner vertices, then \tilde{G} is d -Assur.*

Proof Case 1: \tilde{G} has one inner vertex. \tilde{G} is trivially strongly d -Assur.

Case 2: There is more than one inner vertex. We want to show that \tilde{G} is minimal pinned d -isostatic. If some vertex has valence d , removing it will not cause

the other vertices to move, which contradicts the assumption. Therefore the vertex we remove from \tilde{G} has valence at least $d + 1$.

Since \tilde{G} is a pinned d -isostatic graph, there is a block-triangular matrix decomposition. Assume \tilde{G} is not minimal pinned d -isostatic. Removing any vertex of degree $\geq d + 1$ from any block which is lower right, will leave the upper left block (near the bottom of the directed graph decomposition) as pinned d -isostatic. This guarantees that the solution $R(\tilde{G}, p) \times U = 0$ is zero on all vertices of this upper block (i.e. these vertices have no motion), a contradiction. Therefore \tilde{G} is minimal and d -Assur. ■

In Theorem 7.5.6 it may be surprising that we cannot conclude that the graph will be strongly d -Assur ($d > 2$). Consider the example in Figure 7.18 (a), removing any inner vertex from this graph will cause all other inner vertices to be mobile, yet this is a 3-Assur graph, not strongly 3-Assur.

Theorem 7.5.7 *If we remove any inner vertex from a strongly d -Assur graph \tilde{G} , then at a generic configuration p , \tilde{G} has a finite motion which is non zero at all inner vertices.*

Proof Case 1: \tilde{G} has one inner vertex, removing it leaves no inner vertices.

Case 2: There is more than one inner vertex. Since \tilde{G} is strongly d -Assur, every inner vertex has to be at least valence $d + 1$, as any inner vertex of degree d (and its outgoing edges) in a pinned d -isostatic graph is a strongly d -Assur component (which is called a d -dyad in mechanical engineering). Choose any inner vertex v and remove its edges. As \tilde{G} is strongly d -Assur, removal of these edges, in fact any single edge, will cause a motion at all inner vertices. Now remove the vertex v , all other inner vertices are still in motion. ■

This result is expected, as removal of an inner vertex should cause at least as much motion as a removal of a single edge incident to that vertex. In particular, if

we remove a vertex of degree higher than $d + 1$ there will be more flexibility (more DOF) caused then removal of any edge at that vertex.

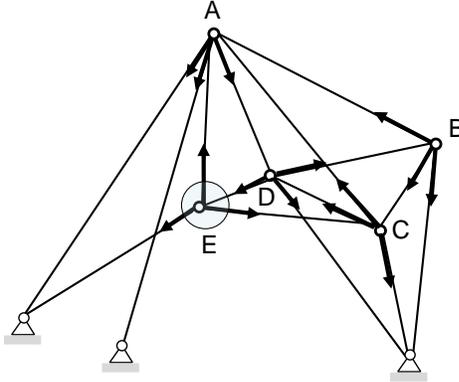


Figure 7.19: A 3-Assur graph, removal of vertex E causes motion in vertices B, C and D, but not in A.

Remark. Most of these concepts and results in sections 7.4 and 7.5 were illustrated using examples in 3-space. All these observations are true in all dimensions. To give higher dimensional examples, one can easily start with a 3-dimensional example and construct similar examples in any higher dimension, using the technique of coning (which essentially adds a new inner vertex and adds an edge from this vertex to all the other vertices in the graph, see [202] for details), which transfers the rigidity, orientation (out-degree), counts, etc. from a framework in dimension d to dimension $d + 1$. ■

7.6 Pinned pebble game algorithm and Assur decomposition algorithm

The work in this section mostly comes from [174]. The use of the pebble game algorithm for studying linkages (mechanisms) and together with the fast efficient

algorithm for decompositions into Assur graphs was for the first time presented to a mechanical engineering community.

In this section we introduce a pinned pebble game algorithm for 2-dimensional (generic) pinned frameworks. We focus on the 2-dimensional pinned frameworks since in dimension 2 we have both necessary and sufficient counting conditions for rigidity (i.e. Pinned Laman’s Theorem 7.4.8). We show how the pinned pebble game algorithm decides if the pinned framework is rigid, the DOF available, and detect any possible redundance. We will not provide a detailed discussion of the pinned pebble game algorithm as most steps and rules are similar to other pebble game algorithms (see Chapter 3). We then use the pinned pebble game algorithm to generate the 2-directed orientation on the graph and to decompose it to 2-Assur graphs.

In dimension $d > 2$, if we assume the graph is pinned d -isostatic, the 2-dimensional Assur decomposition algorithm extends to higher dimensional space. Most of the discussion in this section will be about 2-dimensional pinned graph $\tilde{G} = (I, P; E)$ and throughout we assume the pinned framework realizing the graph is in generic configuration p .

From the Pinned Laman’s Theorem 7.4.8, we can extract this corollary, which is useful to Mechanical engineers [174]:

Corollary 7.6.1 *If a graph \tilde{G} satisfies the modified count: $|E| = 2|I| - k$ for $k > 0$ along with the other inequalities in the Necessary pinned 2-counts (i.e. Pinned Plane Framework Conditions), then generic structures realizing the pinned graph in dimension 2 are k -DOF mechanisms⁸, without redundance.*

Remark. We should mention that mechanical engineers have used counts to predict rigidity and mobility (DOF) of 2-dimensional linkages, primarily relying on the so called Grubler’s counts [66, 72, 163]. However, Grubler’s counts are designed to only

⁸In mechanical engineering, linkages are usually referred to as 1-DOF mechanisms.

check the overall count, so it does not work when there is redundancy in the graph, as it does not check the important subgraph counting conditions [163]. ■

In Figure 7.20 we have several pinned 2-dimensional graphs with various DOF. Once we have introduced the pinned 2-dimensional pebble game algorithm, we will verify the rigidity/DOF of these pinned graphs.

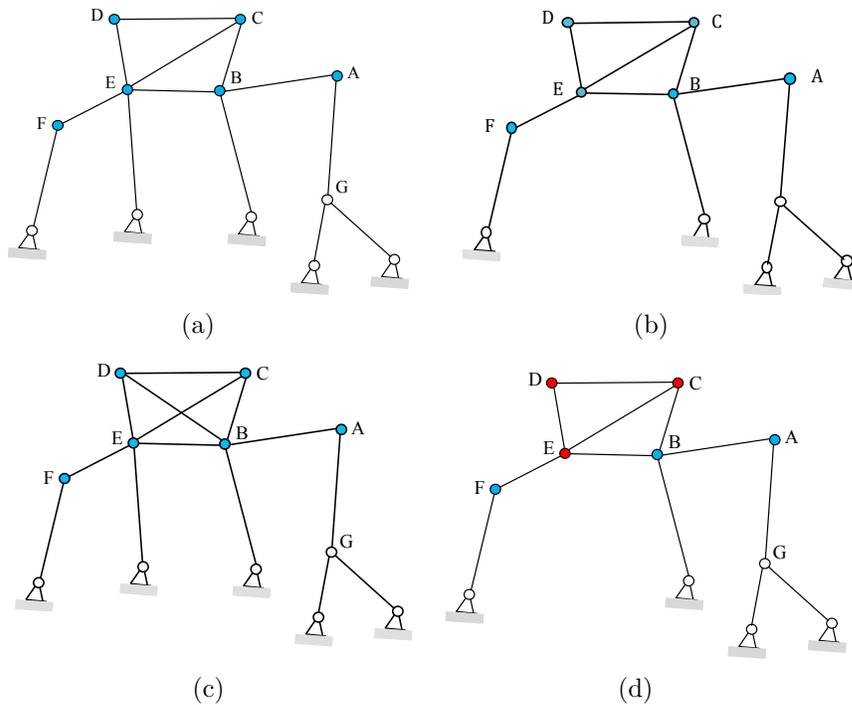


Figure 7.20: Various 2-dimensional pinned graphs. A 1-DOF pinned graph (a) as it is one edge (link, constraint) short of minimum required to be rigid. An isostatic pinned graph (b). A 1-DOF pinned graph with a redundant region (B, C, D, E) (c). A 2-DOF graph (mechanism) (d). Vertex G is not a mobile vertex. Blue vertices have 1-DOF and red vertices have 2-DOF (see Figure 7.22 for output of the pinned pebble game verifying these counts and properties).

7.6.1 Pinned 2-dimensional pebble game

We now provide the pseudocode of the pinned 2-dimensional pebble game algorithm. The pebble game algorithm presented here is designed to check the constraint counting conditions given by the Pinned Laman's Theorem 7.4.8, which we recall here:

1. $|E| = 2|I|$ and
2. for all subgraphs $\tilde{G}' = (I', P'; E')$ the following conditions hold:
 - (i) $|E'| \leq 2|I'|$ if $|P'| \geq 2$,
 - (ii) $|E'| \leq 2|I'| - 1$ if $|P'| = 1$, and
 - (iii) $|E'| \leq 2|I'| - 3$ if $P' = \emptyset$.

As we said earlier most steps and procedures in any pebble game algorithm (i.e. tracking different count) are very similar (see Chapter 3 for details and pebble game terminology in terms of the $6|V| - 6$ pebble game algorithm). However, there are some distinctions in this pinned version of the pebble game algorithm, particularly because of additional counting conditions that need to be satisfied on subgraphs involving pinned vertices, so we need a modified version of the pebble game. This is primarily because we have two types of vertices, inner vertices with 2 DOF and pinned vertices which are not mobile (i.e. pinned vertices have 0 DOF) so they do not get assigned any free pebbles. Our description and development of the pinned version of the pebble game algorithm includes new and additional pebble game checks.

Our pinned version of the pebble game algorithm essentially involves three pebble games that are played on the same graph. That is to say, we need to check three types of counts (i.e. $2|V| - 3$, $2|V| - 1$ and $2|V|$) coming from the pinned Laman's Theorem. This basically means that we are treating the inner edges and ground edges with a different acceptance criteria, because one end of the ground vertex is not mobile (pinned vertex).

Due to the pinned graph setting we have to first play on the inner edges (step 1), which involves the regular (unpinned) $2|V| - 3$ pebble game, and then on the ground edges (step 2), using the $2|V| - 1$ and $2|V|$ games. More specifically, we play on the inner edges first because once a free pebble is placed on the ground edge, that pebble cannot be used in the swap operation as the pinned vertices do not get assigned pebbles. Playing the pebble game in a different order makes no difference to the final outcome of the pebble game as the game is greedy.

Algorithm 7.6.2 – Pinned 2-dimensional Pebble game algorithm:

Input: A pinned graph $\tilde{G} = (I, P; E)$

Output: $D(\tilde{G})$ (set of directed - independent edges) and $R(\tilde{G})$ (set of redundant edges)

Initialize $D(\tilde{G})$ and $R(\tilde{G})$ to empty sets of edges. Place 2 pebbles on each inner vertex of \tilde{G} .

(1.) Test all the non-grounded (inner) edges by playing the $2|V| - 3$ pebble game:

While there exists an untested inner edge e in \tilde{G} :

If endvertices of e (u and v) have 4 free pebbles:

Place a pebble from either u or v onto e , directing e out from that vertex. Place e into $D(\tilde{G})$.

Else: Search for a free pebble from u and v , by following the directed edges in the partially constructed directed graph $D(\tilde{G})$.

If a free pebble is found on some vertex w at the end of the directed path \mathbf{P} : Perform a sequence of swaps (i.e. a cascade - see Chapter 3), reversing the entire path \mathbf{P} , until a free pebble appears on the initial vertex (u or v) of the path \mathbf{P} (w loses one free pebble, and u or v gains one free pebble). Check again for 4 free pebbles.

Else: Place e into $R(\tilde{G})$.

Stop (all inner edges have been tested).

(2.) *Test all the ground edges by playing the $2|V| - 1$ pebble game followed by a $2|V|$ pebble game (if necessary):*

While there is an untested ground edge $e = \{v, w\}$ with inner vertex v and ground vertex w :

Follow the same procedure as in step 1, and play the $2|V| - 1$ pebble game by placing a pebble onto e whenever we have 2 free pebbles on v ⁹. Place e into $D(\tilde{G})$.

Else: We could not recover two free pebbles on the inner vertex v .

If we have only one free pebble on the inner vertex v then place the pebble on e only if the failed search region of e includes at least two distinct pinned vertices (which includes pinned vertex w) (see Figure 7.22 (d))¹⁰. Place e into $D(\tilde{G})$.

Else: e is redundant (not covered by a pebble) and placed in $R(\tilde{G})$ ¹¹.

Stop (all ground edges have been tested).

⁹Recall that the pinned vertex w of the ground edge has no free pebbles, so the maximum free pebbles on the ends of the ground edge will be 2 free pebbles found on the inner vertex v .

¹⁰Note that this is a $2|V|$ pebble game as we can place a pebble on the edge if there is only one free pebble available. The check that the failed search region must include at least two pinned vertices before we can place a pebble onto the ground edge, ensures we maintain the counting condition $|E'| \leq 2|I'| - 1$ if $|P'| = 1$ on all subsets of pebbled edges. If only pinned vertex w is part of the failed search, we cannot pebble edge e as that would violate the $2|I'| - 1$ count. To see this: imagine that we added an isostatic subgraph on the pinned vertices and the pinned vertices are released so now we have an unpinned graph. The isostatic subgraph on these new unpinned vertices will have maximum 3 free pebbles (i.e. 3 trivial DOF of a rigid body). If the failed search region of (ground edge) e contained only one of these unpinned vertices, the ends of e (with two free pebbles on the released pinned vertex) would contain only 3 free pebbles, so we cannot add this edge in the $2|V| - 3$ unpinned pebble game. If on the other hand the failed search region contains two (released) pinned vertices, we can find 4 free pebbles on the ends of edge e and the edge e can be pebbled.

¹¹Note that this will occur if we have no free pebbles on the end of e or if we have one free pebble but the failed search region does not include more than one pinned vertex.

Pinned pebble game by first constructing an unpinned graph

We can describe an alternative version of the pinned 2-dimensional pebble game. We first transform the pinned graph to a unpinned graph by constructing an isostatic (minimally rigid) graph on the pinned vertices.

We first explain the construction process of an unpinned graph to a pinned graph: We construct an isostatic graph on the pinned vertices by adding an edge $e = \{v, w\}$ between two pinned vertices v and w (i.e. $v, w \in P$); it does not matter which two pinned vertices v and w we pick. We take all the remaining (if available) pinned vertices, $u \in P$, and add an edge from u to both ends of v and w (i.e. we get two additional edges $\{u, v\}$ and $\{u, w\}$) (refer back to Figure 7.7 illustrating this construction process).

We then play the $2|V| - 3$ pebble game on the entire unpinned graph (i.e. inner edges, ground edges, including the added extra edges between the pinned vertices). However, we will first play the pebble game on the edges on pinned vertices (i.e. both ends of an edge are pinned vertices) ensuring they all get pebbled. This choice makes no difference to the final outcome of the pebble game as the game is greedy. We continue playing the $2|V| - 3$ pebble game on all edges in $\tilde{G} = (I, P; E)$ (i.e. inner edges and ground edges). Once we are finished the pebble game (i.e. all edges have been tested), we draw back maximum free pebble to the pinned vertices. Pinned vertices will contain exactly 3 free pebbles as it is a rigid subgraph. Then we discard the 3 free pebbles on the pinned vertices (i.e. they are not mobile), along with any added edges between the pinned vertices, which gives us identical output of the pebble game as Algorithm 7.6.2 on the pinned graph. ■

Figure 7.21 depicts the process of applying the Pinned 2-dimensional Pebble Game on the pinned graph of the 1-DOF linkage we initially introduced (Figure 7.3 (b)). Once we have placed 2 free pebbles on each inner vertex, we pick any inner

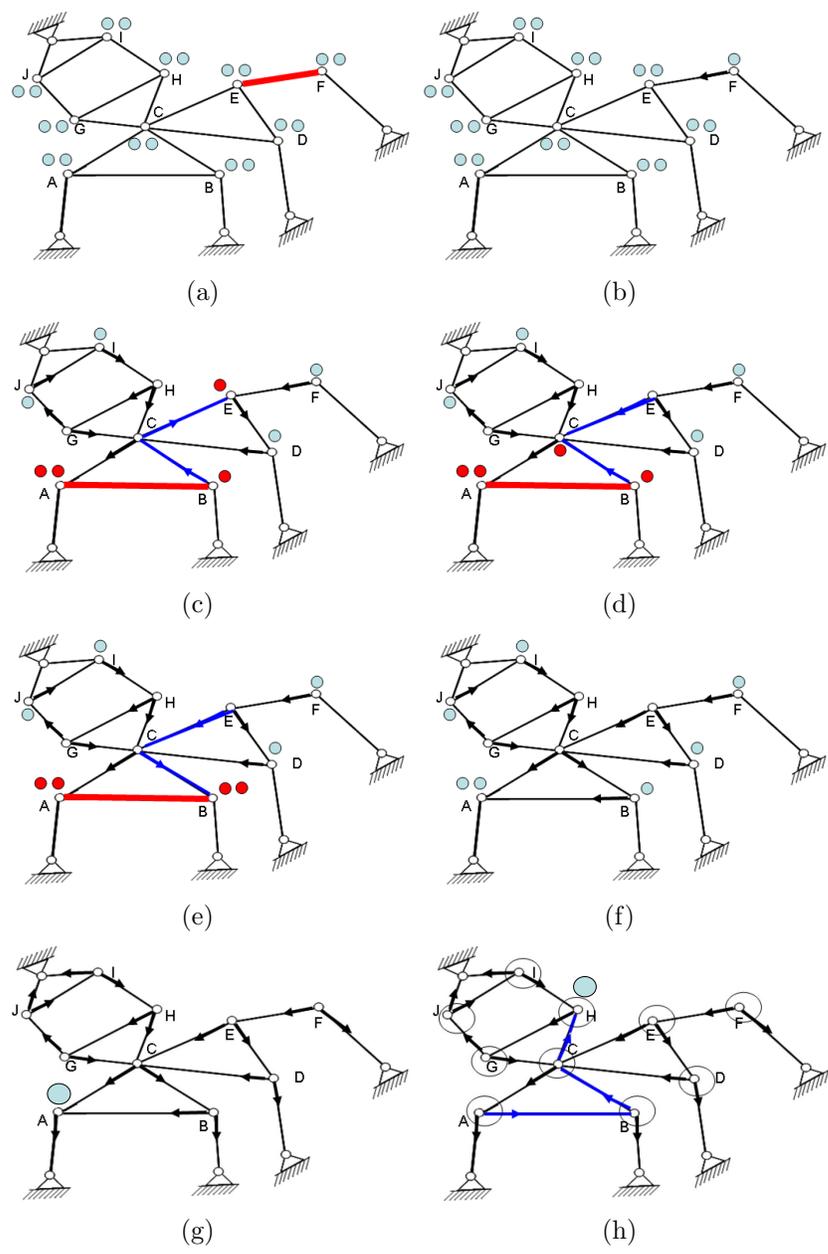


Figure 7.21: Pinned 2-dimensional pebble game (a-g). Inner edges are covered with pebbles when there are four free pebbles on its ends (b). We can search for free pebbles along the directed paths, drawing back the pebble (c, d, e). There is one remaining free pebble, indicating that the corresponding linkage (pinned graph) has 1 DOF, with the pebble placed at the inner vertex of the driver (g). The remaining free pebble can be moved to all the inner vertices along the directed paths, indicating that all vertices are mobile with 1 DOF.

edge, and as long as its ends have four free pebbles we can pebble that edge, directing it out of that vertex (b) (Step 1 of Pinned Pebble game algorithm 7.6.2). When we have four pebbles on the ends, it ensures we respect the critical subtraction $2|I| - 3$ on the subsets of pebbled edges. As with the $6|V| - 6$ pebble game algorithm, when we do not have 2 free pebbles, we can search for a free pebble along the partially constructed directed graph and reverse pebbles using swaps (a cascade) (c-e). Once all the inner vertices have been tested, we pebble all the ground edges (f, g) (Step 2 of algorithm 7.6.2).

The output of the pinned pebble game algorithm 7.6.2 has several key properties which are immediate.

1. The remaining free pebbles tell us how many DOF the pinned graph has. As graph $\tilde{G} = (I, P; E)$ is a pinned graph (i.e. trivial DOF are removed by pinned vertices), any additional free pebbles will be due to the internal DOF.
2. A (generically) rigid pinned graph will have no remaining free pebbles.
3. An isostatic (minimally rigid) pinned graph will have no free pebbles and $D(G) = E$.
4. A mechanism with 1-DOF will have one free pebble.

In our graph in Figure 7.21 there is one free pebble remaining at the end, indicating correctly that this is a 1-DOF mechanism. The remaining free pebble is reversed back to the inner vertex of the driver at the end of the game, which indicates that vertex corresponding to the driver is mobile (Figure 7.21 (g)).

In Figure 7.22 we have played the pebble game on the pinned graphs from Figure 7.20.

We have already seen in Chapters 3 - 5, how distribution of free pebbles and their movement can provide a lot of useful and practical information. In Figure 7.21

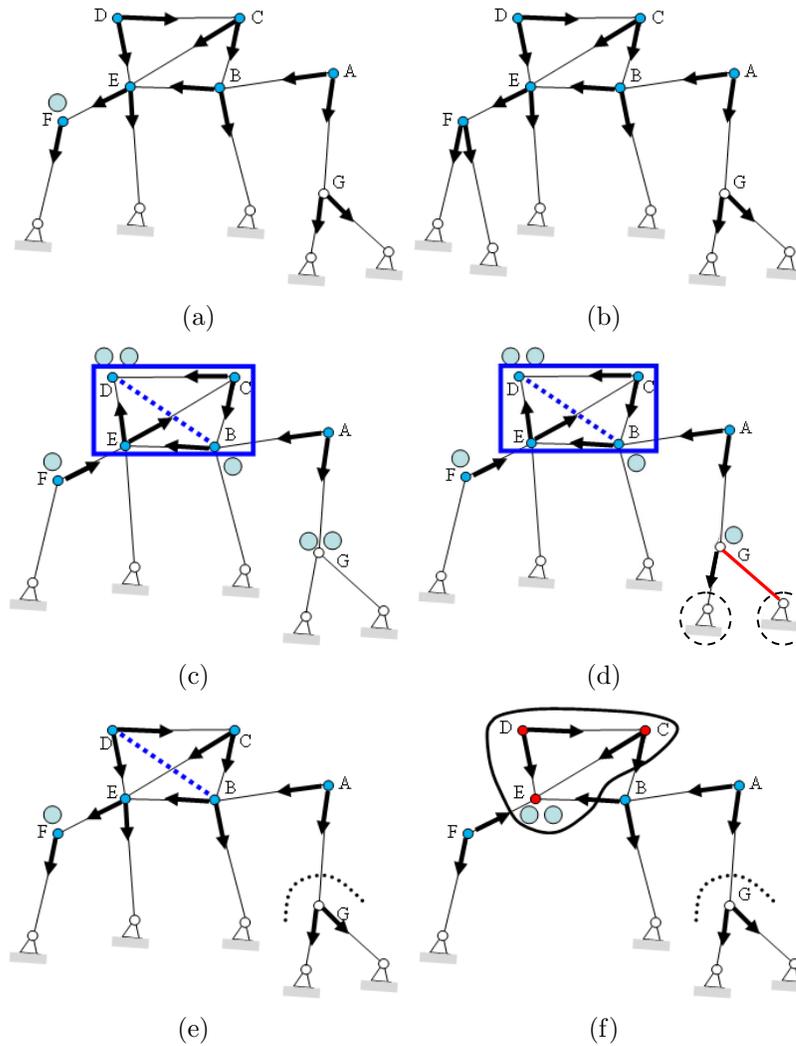


Figure 7.22: Output of the pebble game on 1 DOF pinned graph has 1 free pebble remaining (a), isostatic pinned graph with no pebbles remaining and all edges covered by pebbles (b). When we cannot get the fourth free pebble on the ends of an inner edge (c), the corresponding edge is redundant, and the failed search region (B, C, D, E) is a redundantly rigid subgraph. Any edge could be removed there without affecting the DOF. The pinned graph in (c, d, e) has redundance, but it still has 1 DOF indicated by a remaining free pebble (e). In (d), we have one free pebble on the end of the ground edge (red), which can be pebbled as the failed search region includes at least two pinned vertices (indicated with dotted circles). Joint G is immobile as we cannot get a free pebble there. The graph in (f) has 2 DOF as it has 2 remaining free pebbles. The 2 pebbles are only accessible to joints C, D and E, so only those vertices have 2 DOF and maximum 1 free pebble (1 DOF) to A, B and F.

(h) we see that we can move the one remaining free pebble onto any any inner vertex, as there is a directed path form every inner vertex holding the free pebble. This indicated that all the inner vertices (joints) are mobile with 1-DOF. On the other hand, in Figure 7.22 (a, e, f), the free pebble can never be reversed to joint G, as it is not mobile (vertex G is a simple 2-Assur component, a dyad) and rigidly attached to the ground. In Figure 7.22 (f) the pinned graph has 2 degrees of freedom (2 free pebbles). However, the two free pebbles are only accessible to vertices C, D and E. In other words, we can shuffle the pebbles around and place the 2 pebbles on any one of these three vertices. On the other hand, vertices A, B and F can have a maximum of one free pebble. We will shortly see how tracking free pebbles and their distributions can provide other more useful information.

7.6.2 2-Assur Decomposition Algorithm

If we are given a pinned 2-isostatic graph \tilde{G} (which can be checked with the pinned pebble game algorithm), we can use the pebble game to generate the directions on the edges (i.e. 2-directed orientation) and combined with the strongly connected decomposition of section 7.3 we can then decompose the pinned 2-isostatic graph into 2-Assur components. If we are not provided information (i.e. for instance by mechanical engineer) whether the graph \tilde{G} is pinned 2-isostatic, we can first run the pebble game algorithm to check if \tilde{G} is pinned 2-isostatic and then proceed with the decomposition. In mechanical engineering one may often be given a linkage with say 1-DOF, so in that case we turn it to an isostatic graph and then proceed with the decomposition (see Figure 7.3). We summarize the 2-Assur Decomposition Algorithm using the generated directed graph from the output of the pebble game:

Algorithm 7.6.3 – 2-Assur Decomposition Algorithm:

Given a pinned 2-isostatic graph $\tilde{G} = (I, P; E)$

1. *Generate directions on \tilde{G} towards the ground using the 2-dimensional Pinned Pebble Game Algorithm.*
2. *Find strongly connected components of the directed graph (using Tarjan's algorithm or an equivalent).*
3. *Each strongly connected component and its outgoing edges is a 2-Assur component. For the component make every outgoing edge out of the strongly connected component a ground edge, pinning the end-vertex.*

There are various fast algorithms that can compute the strongly connected components such as Tarjan's [182] whose complexity is $O(|E|)$. The strongly connected decomposition of any directed graph is also included in the core of current computer algebra systems such as Mathematica, Maple and SAGE. The overall complexity of the 2-Assur Decomposition Algorithm is on the order of the complexity of the the pebble game algorithm $O(|V|^2)$ ¹².

In Figure 7.23 we have illustrated the 2-Assur decomposition algorithm 7.6.3 on our original framework. This pinned 2-isostatic graph decomposes into six 2-Assur components, of which 5 are dyads (one inner vertex attached to two pinned vertices).

Remark. We should also stress that the decomposition into 2-Assur components (or in higher dimension) depends on what is declared to be the ground (i.e. pinned vertices) and on the chosen driver. So, for instance in Figure 7.21 (g) if we place the last free pebble of the linkage on another inner vertex (i.e. change the location of the driver) and then add an extra edge from that inner vertex to the ground (or by shifting that inner vertex into a pinned vertex), obtaining a 2-isostatic pinned graph, this will change the directed graph and strongly connected components and consequently the 2-Assur decomposition. So, decomposition is relative to what is a movable piece i.e. what the chosen driver is. However, in mechanical engineering it is

¹²Since typical linkages in mechanical engineering are small (relative to say molecules and proteins), perhaps the complexity of the algorithms is not as crucial.

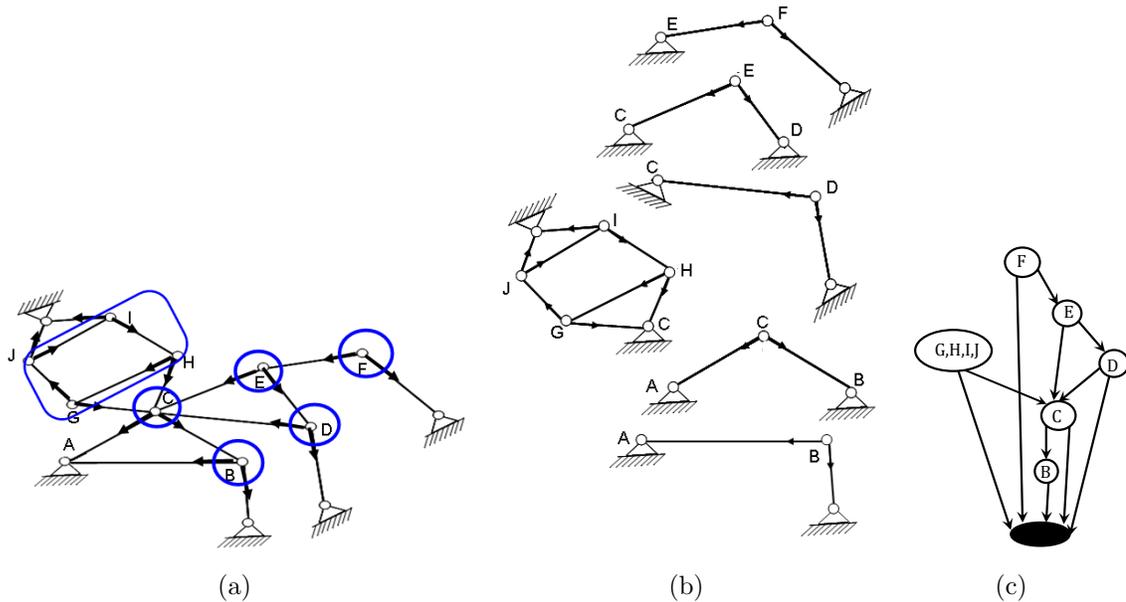


Figure 7.23: The output of the pebble game with generated directions on edges together with strongly connected components (circled in blue) is shown (a, b) (steps 1. and 2. of Algorithm 7.6.3). The individual 2-Assur components (c) (strongly connected components and its outgoing edges forming the ground (pins)) are depicted. The partial order (with all pinned vertices condensed to the ground, and each strongly connected component contracted to a single vertex) is shown (c) (see also Figure 7.4).

usually known where the driver is [174]. Of course, if an alternate pinned 2-isostatic graph is assigned, we can use the pebble game to generate the new directed graph and obtain the decomposition. ■

The pebble game algorithm on a pinned 2-isostatic graph will generate the desired 2-directed orientation (each inner vertex has out-degree 2, and out-degree of each pinned vertex is 0). However, recall that we can have two different generated 2-directed orientations (which differ only up to cycle reversals - Lemmas 7.3.1, 7.3.3) and will still give us the same strongly connected components (Corollary 7.3.4) in the decomposition and consequent uniqueness of the 2-Assur components. We are mainly using the pebble game to generate directions as it is a fast and efficient way to obtain directions. The message here is that it does not matter if we generated directions

from the pebble game or if we were given a set of directions from another source (i.e. mechanical engineer generates it by hand), the decomposition is not changed (see Figure 7.24).

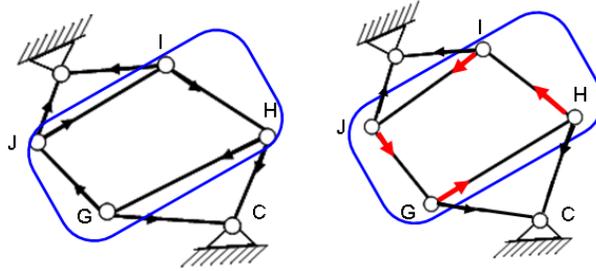


Figure 7.24: Different output of the pebble game can produce a different orientation of the pinned 2-isostatic graph up to cycle reversals (here we have one of the 2-Assur components from Figure 7.23). All inner vertices are still 2-directed, but different orientation (cycle reversal) always leaves the strongly connected components the same, and the subsequent 2-Assur decomposition.

7.6.3 Tracking mobility of vertices using the pebble game

Given a pinned 2-isostatic graph, mechanical engineers are often interested in knowing which vertices will go in motion once some edge is replaced by a driver (i.e. the distance between the ends of the edge is no longer fixed) [164]. In other words, we are interested in predicting which vertices and components will be in motion once some edge is removed from the pinned graph. Will all components move or only a specific collection of components/vertices. This can be answered by looking at the partial order of the decomposition, where any edge removed will cause all the inner vertices in that component to be mobile together with other components that are above in the partial order. Here we want to illustrate this using the output of the pebble game algorithm, which is closely related to our extensions of the pebble game in Chapter 3, which provides us with a visually rich answer to this query, that will be useful to mechanical engineers.

In Figure 7.25 we have illustrated the impact of removal of different edges on the framework we have looked at several times, and which vertices will go in motion. We use the output of the pebble game to track where a free pebble can get to using the directed paths. In (a) removal of blue edge between vertices C and E, leaves an extra free pebble on vertex E (c). This 1 free pebble can only be placed on E or F (i.e. we can swap it back to F) (d), so only these two vertices are going to be in motion (have 1 DOF) . The free pebble is not accessible to other inner vertices, so the rest of the components remain rigid and not in motion. This can also be seen from the partial order of the decomposition (b). In (e) removal of blue edge will cause all components (i.e. all inner vertices) to be in motion, as the free pebble can be reversed along the available directed paths (red). In (g), removal of the edge between C and D, will leave one available free pebble to all inner vertices (i.e. they are mobile) except at inner vertex B.

We give this alternate characterization of 2-Assur graphs using the pinned pebble game:

Theorem 7.6.4 *Let \tilde{G} be a pinned 2-isostatic graph. The following are equivalent:*

1. \tilde{G} is 2-Assur.
2. *The output of the pinned pebble game on \tilde{G} has all edges covered by pebbles and no remaining free pebbles, and removing any edge (ground or inner) leaves 1 free pebble that is accessible¹³ to all inner vertices¹⁴*

Proof \tilde{G} is a pinned 2-isostatic graph, so it will be 2-directed at every inner vertex. All edges in \tilde{G} are covered by a pebble since \tilde{G} is 2-isostatic and as \tilde{G} has 0 DOF, it will have 0 free pebbles. Assume \tilde{G} is 2-Assur, if we remove any edge in \tilde{G} , the free

¹³By accessible we mean that the free pebble that appears from the removed edge on the inner vertex can be drawn back (using a directed path) to any other inner vertex.

¹⁴Note this is also saying that each vertex will be in motion upon removal of any edge, giving us an alternate statement in terms of the pebble game.

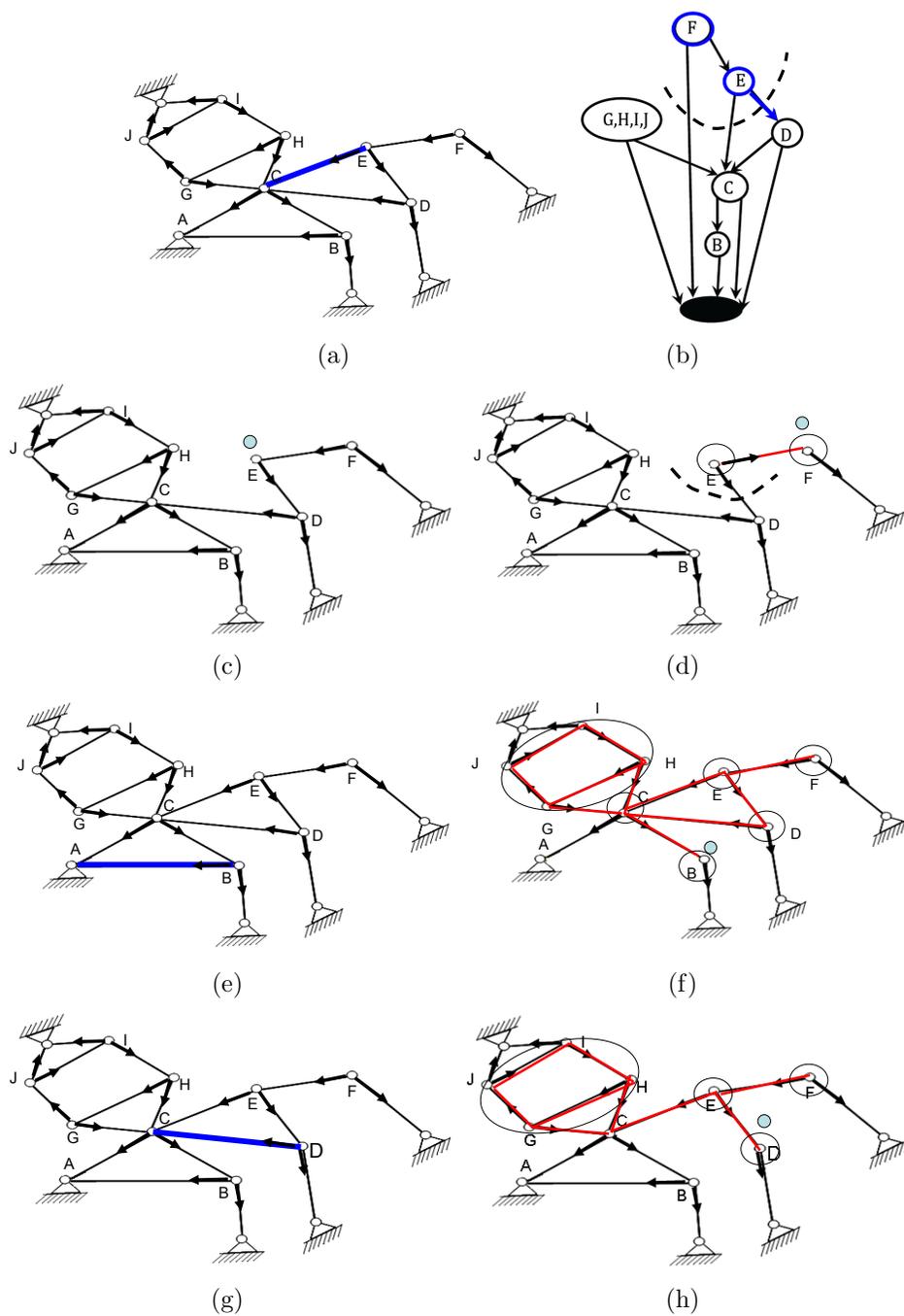


Figure 7.25: Removing an edge from a pinned 2-isostatic graph will cause some (or all) inner vertices to go in motion. Tracking where the free pebble (from the removed edge) can get to gives us an answer. Removal of a blue edge in (a) causes only vertices E and F to be mobile (c, d). In (e, f) all inner vertices are mobile once blue edge is removed, and in (g, h) all inner vertices are mobile except B.

pebble that is removed from this edge will be accessible to all other inner vertices since inner vertices were in a single strongly connected component. More specifically, the edge that is removed only causes its inner vertex with the associated free pebble to have one less outgoing edge. All other set of directed edges are not altered, so we still have a directed path from other vertices to the inner vertex with a free pebble. Conversely, since every inner vertex is reachable from every other inner vertex, the inner vertices are in a single strongly connected component, so \tilde{G} is 2-Assur. ■

Pebble game, Assur decompositions in 3D (and higher)

In this section we were mainly concerned with 2-Assur decompositions, as we have both necessary and sufficient pinned counts for rigidity and we have no distinction between strongly 2-Assur graphs and 2-Assur graphs. As was already discussed, the difficulty in higher dimensional space ($d > 2$) is the lack of counting characterizations of rigidity, so we do not have nice pebble game algorithms that we can apply and check if the given pinned graph is \tilde{G} is d -isostatic.

On the other hand, if we are given a pinned graph that is known to be pinned 3-isostatic (or any $d > 2$, for the sake of applications we concentrate our discussion on 3D), we can still apply a fast pebble game algorithm to generate the 3-directed orientation of the graph, and combined with the strongly connected decomposition, we still have an efficient algorithm for obtaining the 3-Assur components. In this case we would play the $3|V|$ pebble game algorithm on the pinned 3-isostatic graph, which places 3 free pebbles on every inner vertex and places a pebble on an edge (inner edges and ground edges) whenever there exists an available free pebble on the ends of the edge.

Recall that due to the gap in combinatorial (counting) characterizations of rigidity in dimensions higher than 2, once we have obtained the decomposition into

individual 3-Assur components we are still not able to detect which components are strongly 3-Assur. However, since we have the initial decomposition into individual 3-Assur components, this would simplify any further analysis and computations for the mechanical engineer as it is easier to study and analyze the individual 3-Assur components rather than the entire pinned framework.

7.7 Concluding Remarks and Future Work

The decomposition of linkages into minimal fundamental components (i.e. Assur graphs) is an important problem of study in mechanical engineering.

The results in this chapter are contributions to decompositions and analysis of linkages in d -space, and provide several important extensions of 2-dimensional Assur graphs to higher dimensional space. We have presented some additional properties of the Assur decompositions in terms of the lower block-triangular decompositions of the associated pinned rigidity matrix, and we also made some new connections with strongly connected decompositions of directed graphs. This is the first decomposition of Assur graphs in general dimension d (although practically dimensions 2 and 3 are the most important).

We have also highlighted several new connections and difficulties in combinatorial rigidity in higher dimensions ($d > 2$) (i.e. lack of counting characterization of rigidity) of the bar and joint frameworks in terms of the pinned frameworks, and introduced several ‘double-banana’ type of examples. We have also presented some examples of 3-Assur graphs that were not strongly 3-Assur (i.e. removal of some edge(s) does not cause a motion of all inner vertices), which illustrate the distinction of d -Assur graphs and strongly d -Assur graphs that did not occur in the plane (i.e. all 2-Assur graphs are strongly 2-Assur). It is still an open problem to detect combinatorially if a pinned graph in dimension higher than 2 is pinned isostatic, and in

this connection we have also posed a difficult and open problem to combinatorially detect which d -Assur graphs ($d > 2$) are strongly d -Assur.

We have presented an initial transfer of the key pebble game algorithm to the study of linkages to the mechanical engineering community in 2-dimensional linkages, and when combined with strongly connected decomposition it gives us a fast method for obtaining the Assur decompositions. This method also extends to decomposition of pinned isostatic frameworks into Assur components in higher dimensions. We have also demonstrated how the pebble game algorithm can be effectively used to study which vertices go in motion, when some edge is removed from a pinned 2-isostatic framework.

We now highlight some of the future directions in this work and envision other developments.

As we have seen in Chapter 2, body-bar and body-hinge structures have good combinatorial rigidity theory in all dimensions, and corresponding counts that detect if the structure is (generically) pinned isostatic (minimally rigid) (i.e. $6|V| - 6$ count in dimension 3). In the pinned body-bar/body-hinge isostatic structures in dimension 3 we have an underlying pinned multigraph with 6 outgoing edges out of each vertex (i.e. 6 DOF per rigid body). All of the work presented in this chapter directly extends to these alternate structures, and since we do not have the counting problems that are seen in bar and joint structures in dimension 3 and higher, in many ways the extensions and algorithms are easier.

We have nice fast corresponding body-bar/body-hinge, body-pin (i.e. body-hinge in dimension 2) pebble game algorithms (see Chapter 3) that can be applied on the multigraph to test for minimal rigidity and generate the directions on the multigraph. In addition, for body-bar/body-hinge structures all minimal components are strongly d -Assur, so we would not need to make a distinction between d -Assur and

strongly d -Assur (i.e. removal of any edge causes all inner bodies to go in motion) [166]. Moreover the inductive rigidity techniques [203], which can be used to generate Assur components and synthesize new linkages (this was shown for 2-dimensional pinned bar and joint frameworks [162, 163]) are better understood in the body-bar structures in dimension 3 and higher [186] (which are lacking in 3-dimensional bar and joint structures). Given the rich combinatorial rigidity theory of these structures, in the future work, it would be important to investigate body-bar/body-hinge Assur graphs, the available inductive techniques and decomposition algorithms for body-bar/body-hinge Assur graphs. These structures often occur in applications in mechanical engineering and in robotics (for instance Stewart platform is a type of a pinned body-bar structure).

The decompositions that we have presented did not depend on each type of vertex having the same out-degree [166]. They only depended on the out-degree being constant when we switched to another orientation (i.e. under cycle reversals) of the multigraph. So, the techniques and algorithms in this chapter would also be applicable to other mixed structures (i.e. body/bar with some body-hinge structures). With the appropriate application of Assur decompositions to molecular structures, all the decompositions and algorithms would still be applicable.

Given the motivation of important applications to mechanical engineering, the investigations of extensions of Assur decompositions and corresponding algorithms to the body/bar - body/hinge structures are currently underway.

In conclusion, by bringing the methods and algorithms from the rigidity theory, with the work presented in this chapter, a major conceptual advance is attained in understanding the Assur decompositions.

Chapter 8

Conclusion

In the concluding section of key Chapters 4 to 7 we have paved several new research avenues and further future work considerations. Here we offer a few concluding comments.

The overall strategy and objective of this thesis was to develop several algorithms and methods using the concepts from the combinatorial rigidity theory, with the underlying motivation that these algorithms can be used to address important applications and problems in protein flexibility and mechanical linkages.

Using careful extensions of the pebble game algorithm and the FIRST rigid cluster decompositions we are able to accurately predict hinge locations in hinge bending proteins.

The rigidity-based allostery models provide several ways that the two distant sites in a protein can be involved in coupled shape/rigidity changes. The rigidity-based allosteric communication can be analyzed with the corresponding allostery-detection algorithms.

Hinge predictions combined with the rigidity-based allostery models and algorithms, further support and collectively demonstrate the utility and power of the

pebble game algorithm and its extensions. The various pebble game algorithm properties and invariants were also useful in verification of the algorithms and in the proofs of the theorems. The usefulness and power of program FIRST was demonstrated in numerous previous studies, and the rigidity/flexibility predictions on sample proteins were very valuable to us as they formed the inputs to hinge and allostery algorithms. Initial results show that rigidity-based allosteric communication is one possible allosteric mechanism in proteins. Further future work and applications of our algorithms on larger collection of allosteric proteins will reveal how prevalent is the rigidity-based allosteric communication in proteins.

Given the variability in the rigidity and flexibility across the different members of the protein ensemble, we have presented a FIRST-ensemble algorithm. When ensemble rigidity/flexibility prediction is combined with ensemble solvent accessibility it gives a good prediction of the experimentally determined hydrogen/deuterium exchange, as was shown on the case study protein Sso AcP.

Using the pinned pebble game algorithm, together with strongly connected graph decompositions, we have provided an efficient method for decomposing a linkage into important Assur graphs. The initial response from the mechanical engineering community was positive and we anticipate that further future communications and collaborations between the rigidity and the engineering community will be productive to both fields. The natural next extension will be to continue to extend and develop this work to body-bar and body-hinge structures.

References

- [1] S. Ahmad, M. Gromiha, H. Fawareh and A. Sarai, ASAView: Database and tool for solvent accessibility representation in proteins, *BMC Bioinformatics*, 5:51, 2004.
- [2] S. Ahmad, private communication.
- [3] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter, *Molecular Biology of the Cell*, Garland Science, New York, Fourth Edition, 2002.
- [4] B. D. O. Anderson, P. N. Belhumer, T. Eren, A. S. Morse and W. Whiteley, Operations on rigid formations of autonomous agents, *Communications in Information and Systems*, Vol. 3, No. 4, p. 223–258, 2004.
- [5] P.J. Artymiuk, C.C.F. Blake, D.E.P. Grace, S.J. Oatley, D.C. Phillips, M.J.E. Sternberg, Crystallographic studies of the dynamic properties of lysozyme. *Nature* 280:56368, 1979.
- [6] L. Asimov and B. Roth, *The Rigidity Of Graphs*, AMS, 245, 279–289, 1978.
- [7] L. V. Assur. Issledovanie ploskih sterznevnyh mehanizmov s nizsimi parami s tocki zreniya ih struktury i klassikacii. Izdat. Akad. Nauk SSSR, Edited by I. I. Artobolevski. 1952.

- [8] A. Aszodi and W.R. Taylor, Protein Geometry, Classification, Topology and Symmetry, A computational Analysis of Structure, Series in Biophysics, Taylor and Francis, New York, 2005.
- [9] Y. Bai, T.R. Sosnick, L. Mayne and S.W. Englander, Protein folding intermediates: native-state hydrogen exchange, *Science*, Vol. 269 no. 5221 pp. 192-197, 1995.
- [10] G. Barbato, M. Ikura, L.E. Kay, R.W. Pastor, A. Bax, Backbone dynamics of calmodulin studied by I⁵N relaxation using inverse detected NMR spectroscopy: The central helix is flexible. *Biochemistry* 31:5269-5278, 1992.
- [11] M.E. Barnes and W. Whiteley, Preprint, York University, 2005.
- [12] F. Bemporad, J. Gsponer, H.I. Hopearuoho, G. Plakoutsi, G. Stati, M. Stefani, N. Taddei, M. Vendruscolo and F. Chiti, Biological function in a non-native partially folded state of a protein, *The EMBO Journal*, 27, 15251535, 2008.
- [13] The Brookhaven Protein Data Bank (PDB), <http://www.pdb.bnl.gov>
- [14] C.L. Brooks III, M. Karplus, B.M. Pettitt, *Proteins: A Theoretical Perspective of Dynamics, Structure and Thermodynamics*. New York: Wiley, 1988.
- [15] C. Brandon and J. Tooze, *Introduction to Protein Structure*. Garland Publishing, New York, London, 1999.
- [16] D. Bray and T. Duke, CONFORMATIONAL SPREAD: The Propagation of Allosteric States in Large Multiprotein Complexes, *Annu. Rev. Biophys. Biomol. Struct.*, 33:5373, 2004.

- [17] M. Bycroft, S. Ludvigsen, A. R. Fersht, and F. M. Poulsen, Determination of Three-Dimensional Solution Structure of Barnase Using Nuclear Magnetic Resonance Spectroscopy, *Biochemistry* 30, 8697, 1991.
- [18] D.A. Case, Molecular dynamics and normal mode analysis of biomolecular rigidity. In: Thorpe MF, Duxbury PM, eds. *Rigidity Theory and Applications*. New York: Kluwer Academic/Plenum Publishers; 1999. pp 329344.
- [19] J.P. Changeux, F. Jacob and J. Monod, Allosteric proteins and cellular control systems, *J Mol Biol*, 6:306-329, 1963.
- [20] P.A. Chebychev, Theorie des mecanismes connus sous le nom de parallelogrammes, 2eme partie, Momoires presentes a l'Academie imperiale des sciences de Saint Petersburg par divers savants, 1869.
- [21] V. Cherezov, D.M. Rosenbaum, M.A. Hanson, S.G.F. Rasmussen, F.S. Thian, T.S. Kobilka, H.J. Choi, P. Kuhn, W.I. Weis, B.K. Kobilka and R.C. Stevens, High Resolution Crystal Structure of an Engineered Human β_2 -Adrenergic G protein-Coupled Receptor, *Science*, 318(5854): 12581265, 2007.
- [22] S.A. Chervitz, and J.J. Falke, Molecular mechanism of transmembrane signaling by the aspartate receptor: A model, *Proc. Natl. Acad. Sci. USA* Vol. 93, pp. 2545-2550, *Biochemistry*, 1996.
- [23] J, Cheung, W.A. Hendrickson, Structural analysis of ligand stimulation of the histidine kinase NarX, *Structure*, 17(2):190-201, 2009.
- [24] C. Chothia, The nature of the accessible and buried surfaces in proteins, *J. Mol. Biol.* 105, 142, 1976.
- [25] C. Chothia, Principles that determine the structure of proteins, *Ann. Rev. Biochem.*, 53: 537-72, 1984.

- [26] A. Christopoulos and T. Kenakin, G Protein-Coupled Receptor Allostereism and Complexing, *Pharmacological Reviews*, vol. 54 no. 2 323-374, 2002.
- [27] M. Chubynsky, B.M. Hesperheide, D.J. Jacobs, L.A. Kuhn, M.L., S.Menor, A.J. Rader, M.F. Thorpe, W. Whiteley and M.I. Zavodszky, Constraint Theory applied to Proteins. To be published in the proceedings of the Indo-US Biopolymer workshop by Nova Publisher, 2004.
- [28] M.V. Chubynsky and M.F. Thorpe, Algorithms for three-dimensional rigidity analysis and a first-order percolation transition, *Physical Review*, 76, 041135, 2007.
- [29] R. Connely, T. Jordan and W. Whiteley, Generic Global Rigidity of Body-Bar Frameworks, EGRES Technical Report No. 2009-13.
- [30] A. Corazza, C. Rosano, K. Pagano, V. Alverdi, G. Esposito, C. Capanni, F. Bemporad, G. Plakoutsi, M. Stefani, F. Chiti, S. Zuccotti, M. Bolognesi and P. Viglino, P., Structure, Conformational Stability, and Enzymatic Properties of Acylphosphatase From the Hyperthermophile *Sulfolobus solfataricus*, *Proteins*, 62, 64-79, 2006.
- [31] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, *Introduction to Algorithms*, The MIT Press, Cambridge, Massachusetts, Second Edition, 2001.
- [32] H. Crapo, Structural Rigidity, *Structural Topology* 1, 26-45, 1979.
- [33] M.D. Daily and J.J. Gray, Local Motions in a Benchmark of Allosteric Proteins, *Proteins: Structure, Function, and Bioinformatics*, 67:385-399 (2007).
- [34] M.D. Daily and J.J. Gray, Allosteric Communication Occurs via Network of Tertiary and Quaternary Motions in Proteins, *PLoS Comput Biol* 5(2), 2009.

- [35] R.M. Daniel, R.V. Dunn, J.L. Finney and J.C. Smith, The Role of Dynamics in Enzyme Activity, *Annu. Rev. Biophys. Biomol. Struct.*, 32:6992, 2003.
- [36] Database of Macromolecular Movements, <http://www.molmovdb.org>
- [37] B. S. DeDecker, Allosteric Drugs: thinking outside the active-site box, *Chem Biol*, 7:R103-R107, 2000.
- [38] R. Diestel, *Graph Theory*, Springer-Verlag, New York, Second Edition, 2000.
- [39] N.V. Dovidchenko and O.V. Galzitskaya, Prediction of Residue Status to Be Protected or Not Protected From Hydrogen Exchange Using Amino Acid Sequence Only, *The Open Biochemistry Journal*, 2008, 2, 77-80.
- [40] U. Emekli, D. Schneidman-Duhovny, H.J. Wolfson, R. Nussinov and T. Haliloglu, HingeProt: Automated prediction of hinges in protein structures. *Proteins*, 70:1219-1227, 2008.
- [41] S.W. Englander, N.R. Kallenbach, Hydrogen exchange and structural dynamics of proteins and nucleic acids, *Q. Rev. Biophys.* 16 (4): 521-655, 1983.
- [42] S.W. Englander, L. Mayne, Y. Bai, T.R. Sosnick, Hydrogen exchange: the modern legacy of Linderström-Lang, *Protein Sci.* 6, 1101-1109, 1997.
- [43] T. Eren, W. Whiteley, B. D.O. Anderson, A. S. Morse, P. Belhumeur, Information Structures to Secure Control of Rigid Formations with Leader-Follower Structure, In *Proc. of the American Control Conference*, pp. 2966-2971, Portland, Oregon, 2005.
- [44] J.J. Falke and A.H. Erbse, The Piston Rises Again, *Structure*, 17(9): 1149-1151, 2009.

- [45] M. Fiedler, *Special Matrices and Their Applications in Numerical Mathematics*, Springer, 1986.
- [46] W. Finbow-Singh and W. Whiteley, *Isostatic Block and Hole Frameworks*, arXiv:1007.0965v1, 2010.
- [47] G. Fiorin, R.R. Biekofsky, A. Pastore and P. Carloni, *Unwinding the Helical Linker of Calcium-Loaded Calmodulin: A Molecular Dynamics Study*, *PROTEINS: Structure, Function, and Bioinformatics* 61:829839, 2005.
- [48] Flexweb Server (FIRST), <http://flexweb.asu.edu>
- [49] M. Flores, N. Echols, D. Milburn, B.M. Hesperheide, K. Keating, K. Lu, S.A. Wells, E.Z. Yu, M.F. Thorpe and M. Gerstein, *The database of macromolecular motions: new features added at the decade mark*, *Nucl. Acid Res.*, 34 D296301, 2005.
- [50] S.C. Flores and M.B. Gerstein, *FlexOracle: predicting flexible hinges by identification of stable domains*. *BMC Bioinformatics* 8:215, 2007.
- [51] S.C. Flores, L.J. Lu, J. Yang, N. Carriero and M.B Gerstein, *Hinge Atlas: relating protein sequence to sites of structural flexibility*, *BMC Bioinformatics*, 8:167, 2007.
- [52] S.C. Flores, K.S. Keating, J. Painter, Morcos FG, Nguyen K, Merritt E, Kuhn LA, Gerstein M, *HingeMaster: normal mode hinge prediction approach and integration of complementary predictors*. *Proteins*, 73:299319, 2008.
- [53] S.C. Flores, private communication.

- [54] H. Frauenfelder, B.H. McMahon, R.H. Austin, K. Chu, and J.T. Groves, The role of structure, energy landscape, dynamics, and allostery in the enzymatic function of myoglobin, *Proc Natl Acad Sci*;98:23702374, 2001.
- [55] C. Fred E., K.M. Pan, Z. Huang, M. Baldwin, R.J. Fletterick and S.B. Prusiner, Structural Clues to Prion Replication, *Science* 264, 530-31, 1994.
- [56] S. Fulle and H. Gohlke, Analysing the flexibility of RNA structures by constraint counting, *Biophys J* 94:4202-4219, 2008.
- [57] S. Fulle and H. Gohlke, Statics of the ribosomal exit tunnel: Implications for co-translational peptide folding, elongation regulation, and antibiotics binding, *J Mol Biol* 387:502-517, 2009.
- [58] S. Fulle and H. Gohlke, Flexibility analysis of biomacromolecules with application to computer-aided drug design, *Methods Mol Biol.*, 819:75-91, 2012.
- [59] R.A. George and J. Heringa, An analysis of protein domain linkers: their classification and role in protein folding, *Protein Engineering*, 15(11):871-9, 2002.
- [60] M. Gerstein, G. Schulz and C. Chothia, Domain closure in Adenylate Kinase: Joints on either side of two helices close like neighbouring fingers, *J Mol Biol* 229:494501, 1993.
- [61] M. Gerstein. and W. Krebs, A database of macromolecular motions, *Nucleic Acids Res.*, 26, 4280, 1998.
- [62] M. Gerstein, A.M. Lesk, C. Chothia, Structural mechanisms for domain movements, *Biochemistry*, 33:6739-6749, 1994.
- [63] M. Gerstein and C. Chothia, Signal Transduction: Proteins in Motion, *Science*, 285 (5434): 1682-1683, 1999.

- [64] M. Getz, X. Sun, A. Casiano-Negroni, Q. Zhang, and H.M. Al-Hashimi, NMR studies of RNA dynamics and structural plasticity using NMR residual dipolar couplings, *Biopolymers*, 86:384-402, 2007.
- [65] H. Gluck, Almost all simply connected closed surfaces are rigid, In *Geometric Topology*, Lecture notes in Mathematics, Springer-Verlag, Berlin, No. 438, p. 225–239, 1975.
- [66] G. Gogu, Chebychev-Grubler-Kutzbach’s criterion for mobility calculation of multi-loop mechanisms revisited via theory of linear transformations, *European Journal of Mechanics / A Solids*, 24 (3), pg. 427-441, 2005.
- [67] H. Gohlke, L. A. Kuhn, D. A. Case, Change in protein flexibility upon complex formation: Analysis of ras-raf using molecular dynamics and a molecular framework approach, *Proteins*, 56 (2), 322337, 2004.
- [68] N.M. Goodey and S.J. Benkovic, Allosteric regulation and catalysis emerge via a common route, *Nature Chemical Biology*, volume 4 number 8, 2008.
- [69] D. Goodsell and A. Olson, *Structural symmetry and protein function*, *Annu. Rev. Biophys. Biomol. Struct.* **29**, 105–153, 2000.
- [70] J. Graver, B. Servatius, and H. Servatius, *Combinatorial Rigidity*, Graduate Studies in Math., AMS, 1993.
- [71] J. Graver, *Counting on Frameworks: Mathematics to Aid the Design of Rigid Structures*, The Mathematical Association of America, Washington, 2001.
- [72] M. Grubler, Allgemeine Eigenschafter der Zwanglaufigen ebenen kinematischen Ketten, Part I, *Zivilingenieur* 29, pp. 167–200, 1883.

- [73] K. Gunasekaran, B. Ma and R. Nussinov, Is Allostery an Intrinsic Property of all Dynamic Proteins?, *Proteins: Struct., Funct., Bioinf.*, 57, 433443, 2004.
- [74] T.F. Havel, The Role of Tensegrity in Distance Geometry, Rigidity theory and applications, M.F. Thorpe and P.M. Duxbury, Editors, Academic/Kluwer. p. 55-68, 1999.
- [75] S. Hayward, Structural Principles Governing Domain Motions in Proteins, *Proteins*, 36:425-435, 1999.
- [76] S. Hayward and R.A. Lee, Improvements in the analysis of domain motions in proteins from conformational change: DynDom version 1.50. *J Mol Graph Model*, 21(3):181-183, 2002.
- [77] L. Henneberg, *Die graphische Statik der starren Systeme*. Leipzig, 1911; Johnson Reprint, 1968.
- [78] B. Hendrickson, Conditions for unique graph realizations, *SIAM J. Comput.*, 21, pp. 65-84, 1992.
- [79] B. Hendrickson and D.J. Jacobs, An algorithm for two dimensional rigidity percolation: The pebble game, *J. Comput. Phys.*, 137:346-365, 1997.
- [80] J. Hendrickx, B.D.O. Anderson, J-C. Delvenne, and V. Blondel, *International Journal of Robust and Nonlinear Control*, Volume 17, Issue 10-11, 2006.
- [81] J. Heringa, Protein domains, In *Encyclopedia of Genetics, Genomics, Proteomics and Bioinformatics*, Vol. 7, Section 6: Comparative Methods for Structure Analysis and Prediction; Chapter 68, pp. 3283-3297, Wiley Interscience, 2005.

- [82] B.M. Hesperheide, L.A. Kuhn, A.J. Rader and M.F. Thorpe, Identifying protein folding cores from the evolution of flexible regions during unfolding, *J Mol Graph Model*, 21:195-207, 2002.
- [83] B.M. Hesperheide, L.A. Kuhn, A.J. Rader and M.F. Thorpe, Protein unfolding: rigidity lost, *Proc Natl Acad Sci U S A*, 99:3540-5, 2002.
- [84] B.M. Hesperheide, D.J. Jacobs and M.F. Thorpe, Structural rigidity in the capsid assembly of cowpea chlorotic mottle virus, *J. Phys.: Condens. Matter* 16 S5055-64, 2004.
- [85] P. Hu and J.A. Loo, Determining Calcium-binding Stoichiometry and Cooperativity of Parvalbumin and Calmodulin by Mass Spectrometry, *JOURNAL OF MASS SPECTROMETRY*, VOL. 30, 1076-1082, 1995.
- [86] S.J. Hubbard and J.M. Thornton, NACCESS, Department of Biochemistry and Molecular Biology, University College, London, 1993.
- [87] A. Hvidt and S.O. Nielsen, Hydrogen Exchange in Proteins, *Advances in Protein Chemistry*, *Advances in Protein Chemistry*, 21, 287-386, 1996.
- [88] V.P. Jaakola, M.T. Griffith, M.A. Hanson, V. Cherezov, E.Y.T. Chien, J.R. Lane, A.P. Ijzerman and R.C. Stevens, The 2.6 angstrom Crystal Structure of a Human A_{2A} Adenosine Receptor Bound to an Antagonist, *Science*, 322(5905): 12111217, 2008.
- [89] B. Jackson and T. Jordan, Rank and independence in the rigidity matroid of molecular graphs, EGRES Technical Report No. 2006-02.
- [90] B. Jackson and T. Jordan, Rigid components in molecular graphs, EGRES Technical Report No. 2006-03.

- [91] B. Jackson and T. Jordan, The generic rank of body-bar-and-hinge frameworks, EGRES Technical Report No. 2007-06.
- [92] D.J. Jacobs and M.F. Thorpe, Generic rigidity percolation: The pebble game, *Physical Review Letters*, 75:4051-4054, 1995.
- [93] D.J. Jacobs and M.F. Thorpe, Generic rigidity percolation in two dimensions, *Physical Review. E. Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 53:3682-3693, 1996.
- [94] D.J. Jacobs, Generic rigidity in three-dimensional bond-bending networks, *Journal of Physics a-Mathematical and General*, 31:6653-6668, 1998.
- [95] D.J. Jacobs, L.A. Kuhn and M.F. Thorpe, Flexible and rigid regions in proteins, in *Rigidity theory and applications*, M.F. Thorpe and P.M. Duxbury, Editors, Academic/Kluwer. p. 357-384, 1999.
- [96] D.J. Jacobs, L.A. Kuhn, A.J. Rader and M.F. Thorpe, Protein flexibility and dynamics using constraint theory, *J Mol Graph Model*, 19:60-9,2003.
- [97] D.J. Jacobs, private communication.
- [98] Keating, K., Flores, S., Gerstein, M. and Kuhn, L.. StoneHinge: Hinge Prediction by Network Analysis of Individual Protein Structures. *Protein Science* 18: 359-371, 2009.
- [99] J.E. Jimenez-Roldan, R.B. Freedman, R.A. Romer, S.A. Wells, Rapid simulation of protein motion: merging flexibility, rigidity and normal mode analyses, arXiv:1201.5531v1, 2012.
- [100] Katoh, N. and Tanigawa, S., A Proof of the Molecular Conjecture, *Discrete & Computational Geometry* 45: 647-700, 2011.

- [101] W. Kabsch, C. Sander, Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features, *Biopolymers* 22, 2577-2637, 1983.
- [102] L.E. Kavasaki, G.N. Phillips JR. and M.L. Teodoro, *Journal of Computational Biology*, Volume 10, Numbers 34, p. 617-634, 2003.
- [103] A.B. Kempe, How to draw a straight line, (1877) Reprint edition: Classics in Mathematics Education Series, Reston, Va: National Council of Teachers of Mathematics, 1997.
- [104] J. Kim, J. Wess, A. Michiel van Rhee, T. Schoneberg and K.A. Jacobson, Site-directed mutagenesis involved in ligand recognition in the human A_{2A} adenosine receptor, *The Journal of Biological Chemistry*, Vol. 270, No. 23, 12987-12997, 1995.
- [105] T. Knowles and M.J. Buehler, Nanotechnology of functional and pathological amyloid materials, *Nature Nanotechnology*, Vol 6(7), 469-479, 2011.
- [106] Y. Komeiji, Y. Uenob and M. Uebayasia, Molecular dynamics simulations revealed Ca²⁺-dependent conformational change of Calmodulin, *FEBS Letters* 521, 133-139, 2002.
- [107] D.E. Koshland, Protein flexibility in enzyme action and enzyme control, *Science*; 156:540, 1967.
- [108] W.G. Krebs and M. Gerstein, The morph server: a standardized system for analyzing and visualizing macromolecular motions in a database framework. *Nucleic Acids Research*; 28:1665-1675, 2000.
- [109] L.A. Kuhn, D.J. Rader and M.F. Thorpe, Protein flexibility predictions using graph theory, *Proteins*, 44:150-65, 2001.

- [110] ProFlex, <http://www.bch.msu.edu/labs/kuhn/web/software.html>.
- [111] G. Laman, On graphs and rigidity of plane skeletal structures, *J. Eng. Mathematics*, 4:331-340, 1970.
- [112] K. Leach, P.M. Sexton and A. Christopoulos, Allosteric GPCR modulators: taking advantage of permissive receptor pharmacology, *Trends Pharmacol Sci.*, 28(8):382-9, 2007.
- [113] G. Lebon, T. Warne, P.C. Edwards, K. Bennett, C.J. Langmead, A.G.W. Leslie and C.G. Tate, Agonist-bound adenosine A_{2A} receptor structures reveal common features of GPCR activation, *Nature*, Vol 474, 521, 2011.
- [114] A. Lee and I. Streinu. Pebble Game Algorithms and (k, l)-sparse graphs, 2005 European Conference on Combinatorics, Graph Theory and Applications (EuroComb 05), *DMTCS Proceedings*: 181186, 2005.
- [115] A. Lee and I. Streinu. Pebble game algorithms and sparse graphs, *Discrete Mathematics*, 308, 1425–1437, 2008.
- [116] B. Lee and F.M. Richards, The interpretation of protein structures: estimation of static accessibility. *J. Mol. Biol.* 55, 379-400, 1971.
- [117] A.M. Lesk and C. Chothia, Elbow motion in the immunoglobulins involves a molecular ball-and-socket joint, *Nature*: Vol. 335, No. 6186, pp. 188-190, 1988.
- [118] E.J. Levin, D.A. Kondrashov, G.E. Wesenberg and G.N. Phillips Jr., Ensemble refinement of protein crystal structures validation and application, *Structure*, 15(9): 10401052, 2007.
- [119] L. Lins, A. Thomas and R. Brasseur, Analysis of accessible surface of residues in proteins, *Protein Science*, 12:14061417, 2003.

- [120] T. Liu, D. Pantazatos, S. Li, Y. Hamuro, V.J. Hilser and V.L. Jr. Woods, Quantitative Assessment of Protein Structural Models by Comparison of H/D Exchange MS Data with Exchange Behavior Accurately Predicted by DXCOREX, *J. Am. Soc. Mass Spectrom*, 23(1):43-56, 2011.
- [121] G.L. Lukacs, A. Mohamed, N. Kartner, X.B. Chang, J.R. Riordan and S. Grinstein, Conformational maturation of CFTR but not its mutant counterpart (Δ F508) occurs in the endoplasmic reticulum and requires ATP, *EMBO J*, 13(24):60766086, 1994.
- [122] Mamonova T., Hesperheide B., Straub R., Thorpe M.F. and Kurnikova M., Protein flexibility using constraints from molecular dynamics simulations, *Phys. Biol.* 2, S137S147, 2005.
- [123] A. Mantler and J. Snoeyink, Banana Spiders: A study of connectivity in 3D combinatorial rigidity, *CCCG*, 44-47, 2004.
- [124] V. Mathieu, J. Fastrez and P. Soumillion, Engineering allosteric regulation into the hinge region of a circularly permuted TEM-1 Beta-lactamase, *Protein Engineering, Design and Selection*, 23 (9): 699-709, 2010.
- [125] J.C. Maxwell, On reciprocal figures and diagrams of forces, *Phil. Mag.*, 27:250-261, 1864.
- [126] D. Ming, Y. Kong, Y. We and J. Ma, Substructure synthesis method for simulating large molecular complexes *Proc. Natl Acad. Sci. USA* 100 1049, 2003.
- [127] S. Miron, H. Munier-Lehmann, and C. Craescu, Structural and dynamic studies on ligand-free adenylate kinase from *Mycobacterium tuberculosis* revealed a closed conformation that can be related to the reduced catalytic activity, *Biochemistry* 43, 67, 2004.

- [128] S. Mitsi, K.-D. Bouzakis, G. Mansour, and I. Popescu, Position analysis in polynomial form of planar mechanisms with Assur groups of class 3 including revolute and prismatic joints, *Mech. Mach. Theory*, 38(12):1325-1344, 2003.
- [129] A. Momen-Roknabadi, M. Sadeghi, H. Pezeshk and S. Amir Marashi, Impact of residue accessible surface area on the prediction of protein secondary structures, *BMC Bioinformatics* 9:357, 2008.
- [130] C. Moukarzel, An efficient algorithm for testing the generic rigidity of graphs in the plane, *J. Phys. A: Math. Gen.*, 29 8079-98, 1996.
- [131] NAMD, <http://www.ks.uiuc.edu/Research/namd/>
- [132] W.L. Nichols, G.D. Rose, L.F. Ten Eyck and B.H. Zimm, Rigid domains in proteins: An algorithmic approach to their identification. *Proteins*; 23:3848, 1995.
- [133] R. L. Norton, *Design of Machinery: An Introduction To The Synthesis and Analysis of Mechanisms and Machines*. McGraw Hill, New York, 2004.
- [134] R. Nussinov, N. Sinha and C.J. Tsai, Building Blocks, Hinge-Bending Motions and Protein Topology. Volume 19, Issue Number 3, p. 369-380, 2001.
- [135] K.M. Ottemann, W. Xiao, Y.K. Shin, and D.E. Koshland Jr., A Piston Model for Transmembrane Signaling of the Aspartate Receptor, *Science* 285, 1751, 1999.
- [136] J. Owen, Algebraic solution for geometry from dimensional constraints. In *Proc. of the Symp. on Solid Modeling Foundations and CAD/CAM Applications*, 397407, 1991.

- [137] J. Pan and L. Konermann, Calcium-Induced Structural Transitions of the Calmodulin-Melittin System Studied by Electrospray Mass Spectrometry: Conformational Subpopulations and Metal-Unsaturated Intermediates, *Biochemistry*, 49, 34773486, 2010.
- [138] G.R. Pennock and G.M. Kamthe, 2006, A study of dead-center positions of single-degree-of-freedom planar linkages using assur kinematic chains,. *IMEchE J. Mech. Eng. Sci.*, Special Issue on Kinematics, Kinematic Geometry, and their Applications, An Invited Paper, in press.
- [139] A. Perez, A. Noy, F. Lankas, F.J. Luque and M. Orozco, The relative flexibility of B-DNA and A-RNA duplexes: Database analysis, *Nucleic Acids Res*, 2004, 32:6144-6151.
- [140] G.N. Phillips, Describing protein conformational ensembles: beyond static snapshots, *F1000 Biol Reports*, 1:38, 2009.
- [141] C. T. Porter, G. J. Bartlett and J. M. Thornton, *Nucleic Acids Res.*, 32, D129D133, 2004.
- [142] ProFlex, <http://www.bch.msu.edu/~kuhn/software/proflex/>
- [143] PYMOL, <http://pymol.sourceforge.net/>
- [144] S.B. Prusiner, Prion diseases and the BSE crisis. *Science*, 278(5336):245-51, 1997.
- [145] A.J. Rader, G. Anderson, B. Isin, H.G. Khorana, I. Bahar, and Judith Klein-Seetharaman, Identification of core amino acids stabilizing rhodopsin, *PNA*, 72467251, vol. 101, no. 19, 2004.

- [146] A.J. Rader and S.M. Brown, Correlating Allostery with Rigidity, *Molecular Biosystems*, 7, 464471, 2011.
- [147] S. Radestock and H. Gohlke, Exploiting the Link between Protein Rigidity and Thermostability for Data-Driven Protein Engineering, *Eng. Life Sci.* 8, No. 5, 507522, 2008.
- [148] S. Rajagopal and S. Vishveshwara, Short hydrogen bonds in proteins, *FEBS Journal* 272, 18191832, 2005.
- [149] RasMol, <http://www.umass.edu/microbio/rasmol/>
- [150] A. Recki, *Matroid Theory and its Applications*, Springer-Verlag, Berlin, 1989.
- [151] Reduce, <http://kinemage.biochem.duke.edu/software/reduce.php>
- [152] D. Richardson, private communication.
- [153] B. Rost and C. Sander Conservation and prediction of solvent accessibility in protein families. *Proteins*, 20:216226, 1994.
- [154] M. Sadqi, S. Casares, M.A. Abril, O. Lopez-Mayorga, F. Conejero-Lara and E. Freire, The native state conformational ensemble of the SH3 domain from alpha-spectrin., *Biochemistry*, 38(28):8899-906, 1999.
- [155] M. Sammadi, N. Vaidya, A. Sljoka and H. Huaxiong, Coarse graining molecular dynamics with rigidity of hemagglutinin fusion peptides, in preparation.
- [156] K. Schulten and W. Wriggers, Protein Domain Movements: Detection of Rigid Domains and Visualization of Hinges in Comparison of Atomic Coordinates, *Proteins*, 29:1-14, 1997.
- [157] B. Schulze and W. Whiteley The orbit rigidity matrix of a symmetric framework, *Discrete & Computational Geometry*, 46, No. 3, 561–598, 2011.

- [158] B. Schulze, A. Sljoka and W. Whiteley, Protein Flexibility of Dimers: Do Symmetric Motions Play a Role in Allosteric Interactions?, *Advances in Mathematical and Computational Methods: Addressing Modern Challenges of Science, Technology, and Society*, AIP Conference Proceedings, Volume 1368, pp. 135–138, Waterloo, Canada, 2011.
- [159] B. Schulze, A. Sljoka and W. Whiteley, How does symmetry impact the flexibility of proteins?, submitted to *Philosophical Transactions*, Royal Society, 2012.
- [160] W. Scott, C. Schiffer, Curling of flap tips in HIV-1 protease as a mechanism for substrate entry and tolerance of drug resistance, *Struct Fold Design*, 9:1259-1265, 2000.
- [161] B. Servatius and H. Servatius, Generic and Abstract Rigidity, *Rigidity theory and applications*, M.F. Thorpe and P.M. Duxbury, Editors, Academic/Kluwer. p. 1-19, 1999.
- [162] O. Shai, Topological Synthesis of All 2D Mechanisms through Assur Graphs, *ASME Design Engineering Technical Conferences*, Montreal, Quebec, Canada, 2010.
- [163] O. Shai, B. Servatius, and W. Whiteley, Combinatorial characterization of the assur graphs from engineering. *European Journal of Combinatorics* 31, (2010), 1091-1104.
- [164] O. Shai, private communication.
- [165] O. Shai, B. Servatius, and W. Whiteley, Geometric Properties of Assur Graphs. *European Journal of Combinatorics* 31 (2010), 1105-1120.
- [166] O. Shai, A. Sljoka and W. Whiteley Directed graphs, Decompositions and Spatial Rigidity, arXiv:1010.5552, submitted to *Discrete Mathematics*.

- [167] M. Shatsky, R. Nussinov and H.J. Wolfson, Flexible protein alignment and hinge detection. *Proteins*, 48:242256, 2002.
- [168] D.E. Shaw, M.M. Deneroff, R.O. Dror, J.S. Kuskin, R.H. Larson, J.K. Salmon, C. Young, B. Batson, K.J. Bowers, J.C. Chao, M. P. Eastwood, J. Gagliardo, J.P. Grossman, C.R. Ho, D.J. Ierardi, I. Kolossvry, J.L. Klepeis, T. Layman, C. McLeavey, M.A. Moraes, R. Mueller, E.C. Priest, Y. Shan, J. Spengler, M. Theobald, B. Towles, and S. C. Wang, Anton, A Special-Purpose Machine for Molecular Dynamics Simulation, *Communications of the ACM (ACM)* 51 (7): 9197, 2008.
- [169] F.B. Sheinerman and C.L. Brooks III, Molecular picture of folding of a small $\alpha\beta$ protein, *Proc. Natl. Acad. Sci. USA* Vol. 95, pp. 15621567, Biophysics, 1998.
- [170] S.Y. Sheu, E.W. Schlag, H.L. Selzle, and D.Y Yang, Hydrogen Bonds in Membrane Proteins, *J. Phys. Chem.*, 2009.
- [171] A. Shrake and J.A. Rupley, Environment and exposure to solvent of protein atoms. Lysozyme and insulin., *J Mol Biol* 79(2):351-71, 1973.
- [172] A. Sljoka, Counting for Rigidity, Flexibility and extensions via the Pebble Game Algorithm, Masters Thesis, York University, Septemeber 2006.
- [173] A. Sljoka, Counting for Rigidity, Flexibility and Extensions via the Pebble Game Algorithm - Hinge Predictions and other Biological Applications, Third Canadian Student Conference on Biomedical Computing, Paper (2nd prize), 2008.
- [174] A. Sljoka, O. Shai. and W. Whiteley, Checking Mobility and Decomposition of Linkages via Pebble Game Algorithm, Proceedings for ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, IDETC/CIE, Aug 28-31, 2011, Washington, USA, 2011.

- [175] A. Sljoka and A. Bezginov, Predicting Protein Hinge Motions and Allostery Using Rigidity Theory, *Advances in Mathematical and Computational Methods: Addressing Modern Challenges of Science, Technology, and Society*. AIP Conference Proceedings, Volume 1368, pp. 167–170, Waterloo, Canada, 2011.
- [176] A. Sljoka and D. Wilson, Probing Protein Ensemble Rigidity and predictions of Hydrogen-Deuterium exchange using rigidity theory and solvent accessibility, preprint, 2012.
- [177] C.A.E.M. Spronk, S.B. Nabuurs, A.J.J. Bonvinb, E. Krieger, G.W. Vuister and G. Vriend, The precision of NMR structure ensembles revisited, *NMR*, 25: 225234, 2003.
- [178] M.I. Stefan, S.J. Edelstein and N.L. Novere, An allosteric model of calmodulin explains differential activation of PP2B and CaMKII, *PNAS*, 1076810773, vol. 105, no. 31, 2008.
- [179] D. Stewart, A Platform with Six Degrees of Freedom, *Proc. Institution of Mechanical Engineers (UK)*, Vol 180, Pt 1, No 15, 1964.
- [180] K. Sugihara, On redundant bracing in plane skeletal structures, *Bull. Electrotech. Lab.* 44, 376, 1980.
- [181] K. Takedaa, Y. Matsuia, N. Kamiyab, S. Adachib, H. Okumuraa and Tsutomu Kouyama, Crystal Structure of the M Intermediate of Bacteriorhodopsin: Allosteric Structural Changes Mediated by Sliding Movement of a Transmembrane Helix, *Journal of Molecular Biology*, Volume 341, Issue 4, 10231037, 2004.
- [182] R. Tarjan, Depth-first search and linear graph algorithms, *SIAM Journal on Computing*, 1972, Vol. 1, No. 2, P. 146-160.

- [183] S.Y. Tan and M.B. Pepys, Amyloidosis, *Histopathology*, 25, 403-414, 1994.
- [184] T.S. Tay and W. Whiteley, Generating Isostatic Frameworks, *Structural Topology* 11, 21-69, 1985.
- [185] T.S. Tay, Rigidity of Multigraphs I: Linking Rigid Bodies in n-space, *Journal of Combinatorial Theory Series B*, Vol. 26, pp. 95-112, 1984.
- [186] T.S. Tay, Henneberg's method for bar and bod frameworks, *Struct. Topol.* 17, 53-58, 1991.
- [187] M.F. Thorpe, D.J. Jacobs, N.V. Chubynsky and A.J. Rader, Generic rigidity of network glasses. In: Thorpe MF, Duxbury PM, eds. *Rigidity theory and applications*. New York: Kluwer Academic/ Plenum Publishers, pp 239-277, 1999.
- [188] M. F. Thorpe, *Rigidity percolation Physics of Disordered Materials (Institute for Amorphous Studies Series) (New York: Plenum)*, 1985.
- [189] M.F. Thorpe, Protein Folding, HIV and Drug Design *Physics and Technology Forefronts*, APS News, February, 2003.
- [190] C.J. Tsai, A. del Sol and R. Nussinov, Protein allostery, signal transmission and dynamics: a classification scheme of allosteric mechanisms, *Mol. BioSyst.*, 5, 2072-216, 2009.
- [191] W.T. Tutte, On the problem of decomposing a graph into n connected factors. *Journal London Math. Soc.*, 142:221-230, 1961.
- [192] N.K. Vaidya, H. Huang and S. Takagi, Coarse Grained Molecular Dynamics Simulation of Interaction between Hemagglutinin Fusion Peptides and Lipid Bilayer Membranes, *Advances in Applied Mathematics and Mechanics Adv. Appl. Math. Mech.*, Vol., No., pp. 1-21, 2010.

- [193] van der Spoel D, de Groot BL, Hayward S, Berendsen HJ, Vogel HJ, Bending of the calmodulin central helix: a theoretical study, *Protein Sci* 5:20442053, 1996.
- [194] J.P. Vilardaga, M. Frank, C. Krasel, C. Dees, R.A. Nissenson and M.J. Lohse, Differential Conformational Requirements for Activation of G Proteins and the Regulatory Proteins Arrestin and G Protein-coupled Receptor Kinase in the G Protein-coupled Receptor for Parathyroid Hormone (PTH)/PTH-related Protein, *The Journal of Biological Chemistry*, 276, 33435-33443, 2001.
- [195] G. Vriend, WHAT IF, A molecular modelling and drug design program. *J. Mol. Graphics*, 8, 5256, 1990. <http://swift.cmbi.ru.nl/whatif>.
- [196] W. Vranken, A global analysis of NMR distance constraints from the PDB, *J Biomol NMR.*, 39(4): 303314, 2007.
- [197] I.T. Weber and T.A. Steitz, Structure of a complex of catabolite gene activator protein and cyclic AMP refined at 2.5 resolution. *J Mol Biol*;198:311326, 1987.
- [198] S. Wells, private communication.
- [199] S.A. Wells, S. Menor, B.M. Hespeneide and M.F. Thorpe, Constrained geometric simulation of diffusive motion in proteins, *Phys. Biol.*, 2 S12736, 2005.
- [200] S.A. Wells, J.E. Jimenez-Roldan and R.A. Romer, Comparative analysis of rigidity across protein families, *Phys. Biol.* 6, 046005, 2009.
- [201] N. White, *Encyclopedia of Mathematics and its Applications: Theory of Matroids*, Cambridge University Press, Cambridge, 1986.
- [202] W. Whiteley, Cones, infinity and one-story buildings, *Structural Topology* 8, , pp 53 - 70, 1983.

- [203] W. Whiteley, Some Matroids from discrete applied geometry. In J. Bonin, J. Oxley, and B. Servatius, editors, *Matroid Theory*, volume 197 of *Contemp. Math.*, pages 171-311. Amer. Math. Soc., Providence, 1996.
- [204] W. Whiteley, Rigidity of molecular structures: generic and geometric analysis, *Rigidity theory and applications*, M.F. Thorpe and P.M. Duxbury, Editors, Academic/Kluwer. p. 21-46, 1999.
- [205] W. Whiteley, Rigidity and scene analysis, In J. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 60, pages 1327–1354. Chapman Hall/CRC Press, Boca Raton, FL, 2nd edition, 2004.
- [206] W. Whiteley, The Equivalence of Molecular Rigidity Models as Geometric and Generic Graphs, Manuscript, 2004.
- [207] W. Whiteley, Counting out to the flexibility of molecules, *Phys. Biol.* 2, S116-S126, 2005.
- [208] W. Whiteley, private communication.
- [209] M.A. Williams, J.M. Goodfellow and J.M. Thornton, Buried waters and internal cavities in monomeric proteins. *Protein Sci* 3:12241235, 1994.
- [210] D. Wilson, *Conformational Dynamics of Native Proteins Occur Preferentially Along the Folding Reaction Coordinate*, York University, 2008.
- [211] D. Wilson, private communication.
- [212] D. Wishard, Private Communication.
- [213] C. Woodward, I. Simon, et al., Hydrogen-exchange and the Dynamic Structure of Proteins, *Mol. Cell. Chem.*, 48(3): 135-160, 1982.

- [214] Worldwide Protein Data Bank <http://www wwpdb.org>
- [215] F. Xu, H. Wu, V. Katritch, G.W. Han, K.A. Jacobson, Z.G. Gao, V. Cherezov and R.C. Stevens, Structure of an Agonist-Bound Human A2A Adenosine Receptor, *Science* 332, 322, 2011.
- [216] S. Yohannan, S. Faham, D. Yang, J.P. Whitelegge and J.U. Bowie, The evolution of transmembrane helix kinks and the structural diversity of G protein-coupled receptors, *PNAS*, January, vol. 101, no. 4, 959963, 2004.
- [217] M. Zhang, T. Tanaka and M. Ikura, Calcium-induced Conformational Transition revealed by the solution structure of apo Calmodulin, *Nature Structural Biology*, volume 2, number 9, 1995.
- [218] H. Zhang, T. Zhang, K. Chen, S. Shen, J. Ruan, and L. Kurgan, On the relation between residue flexibility and local solvent accessibility in proteins, *Proteins*, 76:617636, 2009.